UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

~ TRABAJO FIN DE GRADO ~

DETECCIÓN DE CAÍDAS PARA VÍDEO-MONITORIZACIÓN EN ENTORNOS DOMÉSTICOS

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Sara Cerro Pardo

Mayo 2014

DETECCIÓN DE CAÍDAS PARA VÍDEO-MONITORIZACIÓN EN ENTORNOS DOMÉSTICOS

AUTOR: Sara Cerro Pardo
TUTOR: José María Martínez Sánchez

Grupo VPULab
Dpto. de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Mayo 2014

"Trabajo parcialmente financiado por el gobierno español bajo el proyecto TEC2011-25995 (EventVideo) "

Agradecimientos

Nunca me había imaginado escribiendo unos agradecimientos y ahora que me he puesto a ello me doy cuenta de que no es una tarea fácil. No porque no haya a quien agradecer, sino porque a veces lo que sientes no puede materializarse en palabras.

En primer lugar, quería dar las gracias de corazón a mi tutor, Chema, por haberme ofrecido esta oportunidad y haber tenido siempre un hueco libre para ofrecerme sus mejores y sabios consejos.

No quisiera olvidarme del resto de profesores de la Escuela, por su cercanía y su esfuerzo en formarnos para llegar hasta aquí. En especial, a Jesús Bescós, por haber estado pendiente de todos nosotros, dispuesto a ofrecer su ayuda ante cualquier problema, pregunta o duda que nos haya podido surgir.

De igual manera, agradecer a todos los compañeros del VPU su disposición para echar siempre una mano, amenizando las mañanas y tardes de trabajo.

Cómo no mencionar a mis compañeros en estos años de carrera. Sin ellos no habría sido lo mismo. Ellos describen lo que es el compañerismo. Los agobios y el estrés continuo no habrían podido superarse sin sus palabras de ánimo y sus bromas, que han hecho que al final sólo me quede con lo bueno. En especial, gracias a Marta y Ana, por ser mi equipo de remo y haberme demostrado su amistad. Y a Alberto, que tiene cabida en varios párrafos aquí, gracias por su ayuda y su buen humor.

A mis amigos de siempre, qué decirles a estas alturas, que aunque a veces son un desastre, siempre están.

Quería dedicar unas líneas a mi gran pasión desde pequeña, el ballet, que me ha hecho evadirme hasta en los momentos más difíciles.

Pero sin duda, gracias a Patricia, mi AMIGA en mayúsculas desde que tengo uso de razón y con quien he tenido la suerte de no despegarme en todo el camino. Porque sé que aunque nunca apuesta, si tuviera que hacerlo, por mí lo haría. Y sin dudarlo, yo por ella.

Y a él, a mi amor, Borja. Que, aunque lejos estos últimos meses, siempre le he sentido muy cerca. No es casualidad que esté escribiendo estos agradecimientos en un día 27.

Y por último, pero lo más importante, a mis padres y a mi hermana Isabel por su apoyo incondicional durante toda mi vida. Sin ellos, su entrega y su amor hoy no sería cómo soy. No alcanzaré nunca a agradecerles el haberme dado tanto y por todas las veces que he salido de casa sin faltarme un "Mucha suerte cariño, que tú vales mucho".

Sara Cerro Pardo
Mayo 2014

Palabras clave

Detección de caídas, sistemas de video-monitorización, segmentación de fondo, representación de centroides, análisis de la figura humana.

Resumen

La motivación principal de este Trabajo Fin de Grado ha sido desarrollar un sistema de detección de caídas, destinado principalmente a aquellas personas mayores que quieran vivir independientes asegurando su bienestar y sintiéndose seguros. Dado el crecimiento constante de este tipo de población, se trata de un sistema con un gran potencial, por lo que se investiga activamente en este campo.

Este sistema de vídeo-monitorización basado en el procesado de vídeo tiene un amplio estado del arte, con numerosas implementaciones y diferentes técnicas usadas para dicho fin.

El objetivo primordial perseguido durante la duración de este trabajo ha sido conseguir unos resultados razonables en la detección de caídas, realizando pruebas con varios vídeos de distintas características y aportando diferentes enfoques con los que establecer conclusiones.

De igual modo, se han comentado los problemas inherentes a este tipo de implementaciones y, más en concreto, aquellas dificultades que han surgido durante este tiempo de trabajo.

Finalmente, se han asentado las bases futuras en las que centrar los esfuerzos para corregir dichos problemas y crear, de esta manera, un algoritmo más robusto que pueda encontrar sitio en este campo tan amplio.

Keywords

Fall detection, vision-based systems, background subtraction, centroid representation, analysis of human shape variation.

Abstract

This work has been motivated by the need to develop a system for fall detection, aimed for senior people who want to live independently assuring their well-being and feeling comfortable. Given the constant growth of this type of population, this system has a high potential, so it is currently being investigated in this field.

This system based on video-monitoring has a large state of the art, with numerous implementations and different techniques used for this purpose.

The main target during the development of this project has been to obtain reasonable results detecting falls, testing the algorithm with datasets made up of videos with different features and providing some approaches to set conclusions.

In the same way, the problems attached to this type of implementations have been explained and, in particular, the difficulties that have appeared throughout this work.

Finally, it has been established the main basis for future studies on how to correct these problems and create at the same time a more robust algorithm so that it can make space for itself in this field.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ECUACIONES

1 Introducción

1.1 *Motivación*

Las caídas en entornos domésticos son una de las causas de mayor riesgo para la población anciana, lo que se convierte en un hándicap para su independencia.
Debido al constante crecimiento de este tipo de población a nivel mundial, el desarrollo de un sistema eficaz que les haga sentir protegidos en este aspecto sería muy importante para su bienestar y su calidad de vida. Este problema se ha incrementado notablemente en los últimos años, causando gran inseguridad en el sujeto, lo que le impide desarrollar sus actividades cotidianas con normalidad. Muchas de estas personas tienen pánico a sufrir caídas y lo que éstas conllevan; como por ejemplo, fracturas de huesos como el de cadera, bastante común en personas mayores y la cual puede llegar a complicarse fácilmente. A veces, incluso puede llegar a causar la muerte, situada ésta como una de las causas más comunes de fallecimiento, directa o indirectamente, entre este tipo de población, produciéndose además el 70% de estas caídas en los hogares [1][2]. Por tanto, encontrar una solución a este problema es uno de los principales retos para la sociedad de hoy en día.

Como veremos en el estado del arte, existen muchas vías posibles para dar solución a este problema, en lo que respecta a distintos mecanismos y dispositivos que ayuden a detectar la caída e informar a los familiares sobre el suceso. Sin embargo, no todas son aceptadas por la persona que requiere esta protección, ya que a algunas les pueden resultar intrusivas, molestas o proclives a ser olvidadas, como sucede en el caso de dispositivos que son necesarios llevar puestos. Por estas y otras razones, no todas las soluciones cumplen con su cometido y son lo suficientemente efectivas.

Como resultado de estas reflexiones, la motivación principal de este proyecto será diseñar e implementar un sistema basado en el procesado de vídeo que detecte caídas simuladas en entornos similares a uno doméstico y suponiendo que funcionará de igual modo en caso de caídas reales, ya que no se dispone de vídeos para evaluación en dicha situación.

1.2 *Objetivos*

Para abordar este objetivo, se enfocará el trabajo de manera progresiva, cumpliendo objetivos parciales hasta llegar al objetivo global:

- **Estudio del estado del arte:** Se analizarán los distintos métodos y sistemas que existen actualmente para la detección de caídas, comparando las propiedades que presentan bajo diversas condiciones y eligiendo, finalmente, una de las aproximaciones para desarrollar la propia implementación.
- **Análisis en profundidad del método seleccionado:** Una vez seleccionado el sistema que se considere más conveniente tanto por sus resultados como por su posibilidad en el desarrollo, se analizará en detalle dicho método y posibles mejoras al mismo.

- **Desarrollo del algoritmo:** Desarrollo del método seleccionado buscando, en primera instancia, obtener los mismos resultados, para más tarde incorporar las mejoras seleccionadas.
- **Evaluación de resultados:** Se pretende medir la calidad del sistema generado, analizando los resultados obtenidos para evaluar el rendimiento del sistema, aportando varios enfoques y comparándolos entre ellos.

De este modo, se concluye que el objetivo fundamental de este trabajo es reproducir los resultados obtenidos en el método seleccionado, incluyendo nuevas opciones no contempladas en su diseño original.

1.3 *Organización de la memoria*

La memoria consta de los siguientes capítulos:

- **Capítulo 1: Introducción.** Motivación, objetivos y estructura de la memoria del proyecto.
- **Capítulo 2: Estado del arte**. Estudio de los principales métodos y dispositivos de detección de caídas, enfocados a su aplicación en entornos domésticos. Análisis comparativo de las ventajas y desventajas de cada uno de ellos y recopilación de los *datasets*[1] utilizados para cada implementación analizada.
- **Capítulo 3: Diseño y desarrollo del sistema.** Descripción tanto del sistema a reproducir como del posteriormente desarrollado, detallando las técnicas y herramientas utilizadas para conseguir detectar caídas simuladas.
- **Capítulo 4: Evaluación del sistema.** Evaluación del algoritmo implementado mediante un *dataset* de detección de caídas. Así, se busca verificar el correcto funcionamiento del sistema, basado en algoritmos en dos dimensiones. Análisis de resultados del método propio con varios enfoques, contrastando los distintos métodos.
- **Capítulo 5: Conclusiones y trabajo futuro.** Conclusiones obtenidas tras el análisis de resultados. Problemas pendientes, posibles mejoras y líneas de trabajo futuras.
- **Referencias y anexos.**

[1] Conjunto de datos (e.g. vídeos) para evaluación de algoritmos, así como su *ground truth* asociado.

2 Estado del arte

2.1 *Introducción*

Cada año miles de personas mayores son víctimas de caídas en su hogar. Esto es así, ya que a medida que una persona se va haciendo mayor, su cuerpo es más frágil y el riesgo de sufrir accidentes como caídas aumenta notablemente.

De la caída en sí pueden derivar otras consecuencias que incrementen la gravedad del suceso. Estas consecuencias dadas tras la caída pueden ser la inmovilidad del sujeto durante largas horas e incluso días en el suelo de su hogar, tanto inconscientes como despiertos, pero incapaces de poder incorporarse de nuevo o pedir ayuda, lo que puede conllevar más daños tales como hipotermia, al permanecer tendidos en el suelo frío durante largos períodos de tiempo.

Por ello y, teniendo en cuenta la mayor población de personas ancianas que existe hoy en día y, que va en aumento, es necesario establecer sistemas de protección para este propósito.

2.2 *Tipos de métodos de detección de caídas*

Desde que se empezó a investigar y a buscar soluciones para este problema, se han desarrollado muy variadas y diferentes técnicas para la detección de caídas. Por ello, se ha establecido una jerarquía de métodos de detección en tres categorías diferentes [2]:

- Métodos basados en dispositivos portátiles.
- Métodos basados en dispositivos en el entorno.
- Métodos basados en el análisis de vídeo.

A continuación se detallan estos métodos.

2.2.1 Métodos basados en dispositivos portátiles

Este enfoque se basa en la utilización de sensores que se colocan en las prendas de ropa de las personas proclives a caídas, pudiendo detectar el movimiento y la localización del cuerpo. Se proponen diferentes métodos, la mayoría basados en acelerómetros [2][3], instrumentos que miden aceleraciones de una masa de prueba que se encuentra en el marco de referencia del dispositivo.

El uso de acelerómetros es un método extensible a muchas combinaciones y diferentes técnicas, todas orientadas a medir la actividad del cuerpo o de partes del cuerpo. Pueden integrarse en la cintura del sujeto [2], detectando caída al aumentar la aceleración negativa del mismo por un cambio repentino, por ejemplo de estar de pie a estar tumbado. Una mejor opción es que estos acelerómetros tengan incorporados sensores que midan la presión barométrica [2]. Otra posibilidad consiste en introducir un airbag que se infle al

superar unos umbrales en la aceleración y velocidad angular [2], reduciendo así el riesgo de daño en la caída.

Por otro lado, tenemos que muchas de las respuestas fisiológicas, como el cambio de presión sanguínea, la tasa de respiración o las pulsaciones del corazón, están íntimamente relacionadas con el cambio de posición. Esta es la clave para otras tantas implementaciones con este tipo de dispositivos.

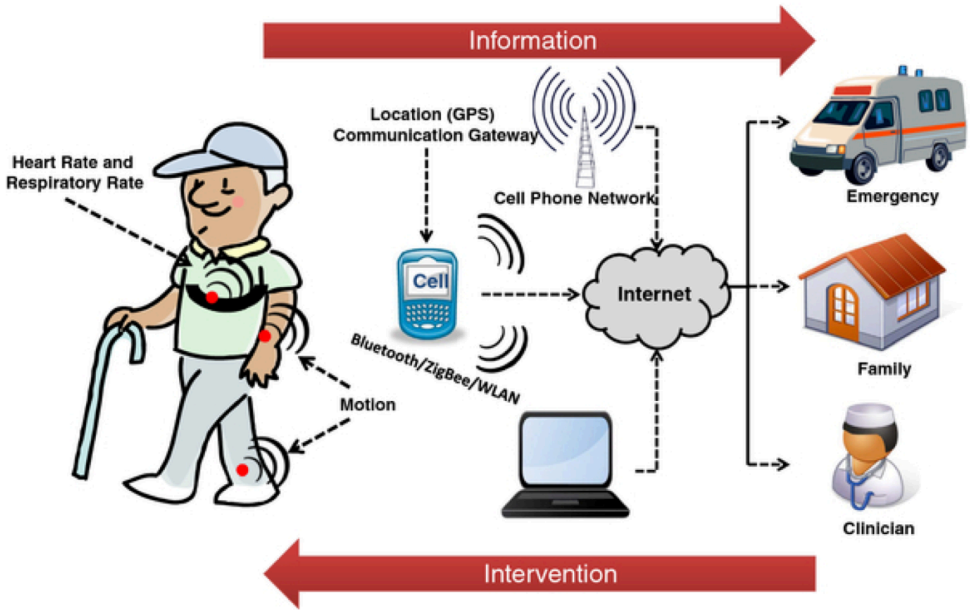

Figura 2-1: Sistemas de vigilancia basados en sensores portátiles. Fuente: [5]

Es importante saber medir e identificar cuándo el sujeto está inactivo, es decir, en ausencia de movimiento, ya que puede ofrecernos mucha información, la cual servirá para analizar características del movimiento como la frecuencia, la intensidad y la duración existentes durante el mismo. Un diseño basado en este principio es SIMBAD (*Smart Inactivity Monitor using Array-Based Detectors*) [2].

Si en vez de utilizar un solo *array* de detectores infrarrojos, incorporamos acelerómetros tri-axiales [2] que midan la aceleración en tres direcciones axiales, seremos capaces de detectar conjuntamente las posibles partes del cuerpo lesionadas en la caída. Y si, además, incorporamos acelerómetros multicanal [2] que se basen en mecanismos posturales y de orientación podremos distinguir anomalías en los patrones de movimiento clásicos. Estos dispositivos son pequeños y se colocan en distintas y significativas partes del cuerpo como cadera, cintura, rodillas, hombros, codos, tobillos, etc., que servirán para reconstruir el cuerpo en un plano 3D. Esta idea de reconstrucción es muy útil para obtener información sobre la caída y, por tanto, de las partes que han podido ser dañadas.

2.2.1.1 → Ventajas e inconvenientes.

La mayor ventaja es el coste de estos dispositivos, los cuales son bastante asequibles. Además, la instalación, configuración y operación son relativamente fáciles y no suelen resultar problemáticas.

Por el contrario, estos dispositivos pueden resultar incómodos para la persona que debe llevarlos encima. Junto a esta incomodidad, se une la preocupación de recargar sus baterías continuamente [4], así como estar pendientes de que no se desconecten, lo que ocurre a menudo al tener que realizar las tareas diarias portando este tipo de aparatos. Igualmente, muchos requieren la intervención de la propia víctima [4], teniendo que apretar un botón en caso de caída, lo que en caso de caer inconsciente no será posible.
Por otro lado, estos mecanismos diariamente les recuerdan que son vulnerables, lo que deriva en un factor psicológico muy perjudicial para ellos.

En conclusión, vemos que los dispositivos portátiles, aunque compactos, sencillos y relativamente baratos, no son una opción muy favorable, tanto por interferir notablemente en el bienestar del sujeto, como por la baja fiabilidad que resulta de los problemas ya citados.

2.2.2 Métodos basados en dispositivos en el entorno

Estos dispositivos tratan de fusionar el audio, vídeo y, opcionalmente, datos de vibraciones para la detección de caídas.
Se pretende crear un puente entre el usuario y la red mediante una placa de nodos inalámbricos que detectan caídas a través de funciones de detección de eventos[2]. Si además se crea una comunicación de voz entre el usuario y el monitor de control, es posible que cuando ocurre la caída y el sistema la detecte, se active una alarma.
La incorporación de datos de vibraciones [2][5] es muy útil a la hora de monitorizar, seguir y localizar las caídas. Por ejemplo, una implementación según este diseño es un detector de caídas basado en los patrones de vibración del suelo [2].

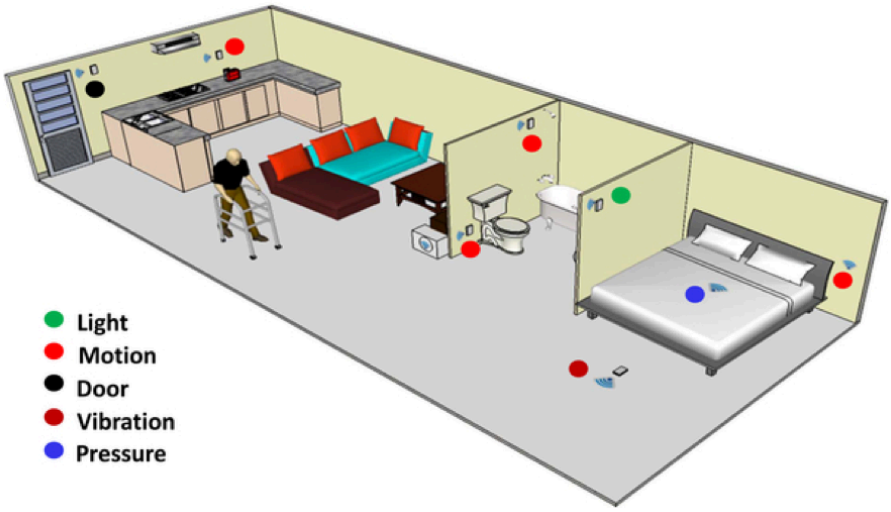

Figura 2-2: Incorporación de sensores en el hogar. Fuente: [5]

La idea que sigue este método se fundamenta en que los patrones de vibración que se producen en el suelo en una caída son muy distintos de los producidos por otros eventos como andar o estar de pie. El procedimiento es el siguiente: se acopla una pieza eléctrica en el suelo, así como un circuito de tratamiento previo cargado por una batería y empleado

para analizar los patrones de la señal de caída. Cuando tenga lugar dicha caída, se puede generar una señal binaria que registre dicho evento.
El sistema es muy útil en casos críticos tales como cuando el sujeto se encuentra inconsciente o en estado de shock.

2.2.2.1 → Ventajas e inconvenientes

Muchos de los dispositivos de ambiente utilizan sensores de presión para la detección y el seguimiento de caídas, basándose en el principio de alta presión. Este método es rentable y menos intrusivo que las implementaciones de sistemas de vigilancia.
Sin embargo, una gran desventaja es que, al estar constantemente midiendo la presión del ambiente alrededor del sujeto, se pueden generar falsas alarmas, por lo que la precisión y fiabilidad no están totalmente aseguradas.

2.2.3 Métodos basados en el análisis de vídeo

El sistema basado en la incorporación de cámaras ha crecido notablemente en los últimos tiempos como método de asistencia para la detección de caídas, ya que posee muchas más ventajas que los otros métodos citados hasta el momento. Además, se ha visto que un sistema de video-monitorización puede ser utilizado para detectar múltiples eventos al mismo tiempo; no sólo caídas, aunque este es el fin que perseguimos en este trabajo.

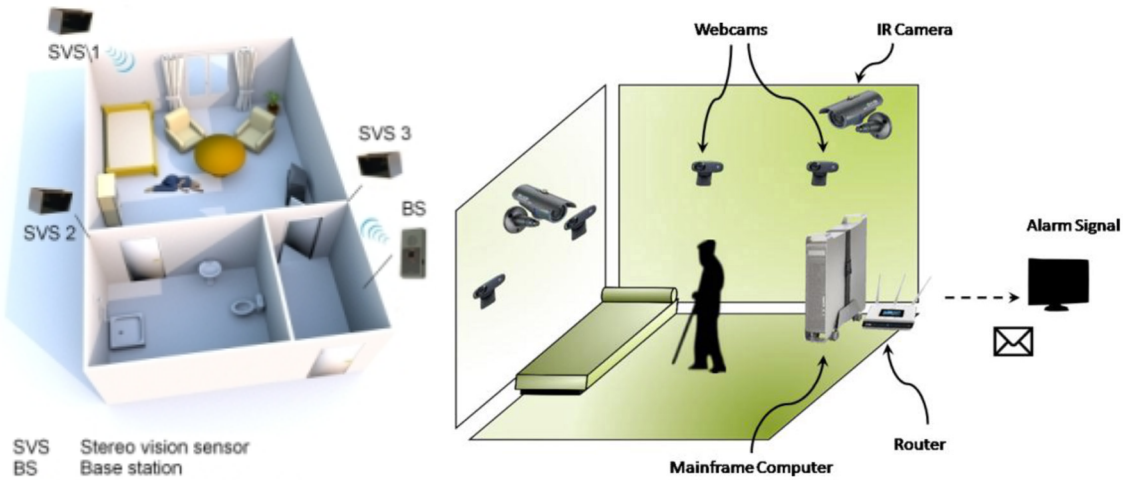

Figura 2-3: Sistema de video-monitorización por análisis de vídeo. Fuente: *Google* imágenes.

2.2.3.1 → Ventajas e inconvenientes.

Las principales ventajas de este método son su baja intrusión y la poca o nula interacción que debe tener el sujeto con el sistema, ya que no tiene que preocuparse de ponerse ni configurar ningún dispositivo, así como ni de estar pendiente de recargar sus baterías u otras consideraciones a tener en cuenta con los métodos mencionados anteriormente. Además, resulta un método muy robusto y eficaz, que aunque requiere de una instalación previa, ésta no es demasiado costosa tanto a nivel económico como de impacto en el hogar.

2.2.4 Comparativa y conclusiones

A continuación se muestra una tabla comparativa de las técnicas mencionadas y sus principales características, vistas como posibles ventajas o inconvenientes:

Enfoque	Coste	Intrusión	Precisión	Configuración	Robustez
Dispositivos portátiles	Barato	Sí	Dependiente del escenario	Fácil	No
Dispositivos en el ambiente	Barato/ Medio	Sí	Dependiente del escenario	Fácil/Media	No
Sistemas de cámaras	Medio	Baja (dependiente del sistema)	Alta	Media	Sí

Tabla 2-1: Comparativa de métodos de detección de caídas. Fuente: Propia.

Tras la anterior comparación, parece obvio centrar nuestros esfuerzos en la creación de un sistema robusto basado en el procesado de vídeo para la detección de eventos sospechosos como las caídas en entornos domésticos. Dada su gran eficacia, se va a hacer hincapié en las técnicas basadas en dicho procesamiento.

Como ya se ha hecho referencia, existen numerosas técnicas y algoritmos diferentes que obtienen resultados aceptables utilizando este método. Si bien, algunas tienen un porcentaje mayor de aciertos que sus competidoras; otras, en cambio, poseen una menor intrusión, pieza clave en la privacidad del sujeto.

2.3 *Algoritmos de detección de caídas basados en análisis de vídeo*

2.3.1 Introducción

Todos los sistemas basados en el procesado de vídeo siguen una estructura similar [3]: el primer paso es un modelado de fondo que determine y localice el movimiento del sujeto. Una vez ha sido detectado el evento de interés, se procede a extraer diferentes características que indicien cualquier tipo de caída. Tras dicha detección, será necesario generar una alarma que alerte del suceso producido, aunque este último posible paso no se contempla en este trabajo.

A continuación, se hace una clasificación de las técnicas que existen:

- Algoritmos 2D.
- Algoritmos 3D.
- Algoritmos 2,5D (también conocidos como *2D+depth* o *RGB+depth*).

2.3.2 Caracterización de algoritmos de detección de caídas

Para desarrollar un método efectivo de detección de caída es esencial entender bien el concepto de caída, así como las distintas características en cada tipo de caída. En general,

una caída se detecta cuando una persona cambia de una postura erguida y vertical a una horizontal de manera rápida y, seguidamente, permanece en el suelo inmóvil durante un tiempo a causa del impacto de la caída.
Existen diferentes enfoques para realizar un algoritmo de detección de caídas. Algunos destacan por su simplicidad y aceptable eficacia, por lo que son ampliamente utilizados. Otros más innovadores, se centran en distintos aspectos que veremos a continuación, los cuales resultan en técnicas muy eficientes que a día de hoy están en pleno desarrollo.

Hay ciertas consideraciones que deben tenerse en cuenta con un sistema de estas características, mencionadas en [3][4]:
Por un lado, el tema de la privacidad es uno de los más controvertidos. Las personas que dispongan de este sistema deben dar su consentimiento para instalar las cámaras en sus hogares, a la vez que deben ser plenamente conscientes de que están siendo grabados, aunque nadie esté observando aquello que se graba, ya que simplemente estará siendo procesado en un ordenador. Sin embargo, las cámaras grabarán escenas en situaciones privadas, como en el baño, ya que desgraciadamente éste constituye uno de los principales lugares de mayor riesgo de caída en el hogar. Situar las cámaras en lugares discretos contribuirá al sentimiento de comodidad de la persona a ser grabada.
Por otro, y centrándonos más en las dificultades que se dan al desarrollar el algoritmo, tenemos las siguientes:

- Problemas con la iluminación y sombras que aparecen en la imagen, que pueden dar problemas en el proceso de segmentación.
- Problemas con la movilidad de objetos y muebles dentro del escenario de grabación. Este detalle debe tenerse en cuenta para que el modelado de fondo se actualice asiduamente, de modo que dichos cambios no afecten a la eficacia de nuestro sistema.
- Problemas de oclusión con objetos, totales o parciales.
- Problemas debidos a la entrada/salida del campo de visión de la cámara.
- Problemas surgidos por la diferente ropa del sujeto, en cuanto a colores y texturas, así como en situaciones de ponerse/quitarse la misma.

Una vez establecidas las consideraciones generales que deben tenerse en mente a la hora de desarrollar un algoritmo de este tipo, se pasa a analizar en profundidad cada uno de ellos:

2.3.3 Algoritmos 2D

Se trata de los algoritmos más sencillos que utilizan el procesado de vídeo y en los cuales sólo se requiere de una cámara para la grabación, por lo que es un sistema que no necesita calibración. Son fáciles de implementar y proporcionan unos resultados bastante razonables, a pesar de que ciertos problemas tales como la oclusión de objetos o movimientos similares a caídas que pueden llevar a confusión, no siempre se resuelvan con éxito.
A continuación describimos los más utilizados:

2.3.3.1 Análisis de la silueta humana

Se puede llevar a cabo de múltiples formas. Vamos a ver algunas de ellas, que utilizan:

- **Bounding-box** (mencionado en [2][3][4][6][7]).

Se calcula como una razón/ratio entre la altura y la anchura del objeto "encerrado" por dicha caja, analizándola para cada *frame*[2] y comparando con sus anteriores. Hay que tener en cuenta que cada persona tendrá unos ratios diferentes entre el estado normal y el de caída, por lo que puede utilizarse como otro parámetro útil a tener en cuenta el Índice de Masa Corporal, que servirá para establecer un umbral.

Es un método simple que funciona bien y tiene un bajo coste computacional. Sin embargo, solo es válido y efectivo cuando la cámara está situada oblicuamente. La eficiencia de este método depende de la posición relativa de la persona y del campo de visión del que dispone la cámara, tendiendo a fallar cuando se presentan situaciones de oclusión.

- **Elipse aproximada alrededor de la silueta** (mencionado en [2]).

Se trata de una elipse que describe la silueta del sujeto, ajustándose lo más posible a ella. Es necesario utilizarla en conjunto con otras técnicas, como el análisis de movimiento o la estimación de la postura del sujeto, para poder obtener buenos resultados.
Una de las posibles implementaciones, consiste en la segmentación del sujeto en movimiento como paso inicial, para después realizar un histograma donde obtener las características extraídas del análisis del cambio de silueta. Se termina con la detección de movimiento temporal, incluyendo una información adicional al incorporar aquella extraída de la posición de la cabeza.
En la mayoría de los casos constituye una técnica que suele dar mejores resultados que las que utilizan *bounding-box*. Sin embargo, obtiene fallos en actividades como sentarse bruscamente, agacharse, etc., que dan lugar a falsos positivos.

- **Extracción de tres puntos (cabeza, cuerpo y piernas) y análisis de sus variaciones** (mencionado en [7]).

Se trata de una técnica basada en la variación de la silueta humana, pero usando solo tres puntos: los centroides correspondientes a cabeza, cuerpo y piernas. De este modo, es posible representar a la persona. Se demuestra que mediante este procedimiento no sólo se aumenta la tasa de detección de caídas, sino que también se reduce el coste computacional.

[2] Cada una de las imágenes en las que se descompone un vídeo.

Una de las implementaciones que siguen esta técnica lleva a cabo el proceso de la siguiente manera: con estos tres puntos extraídos de la silueta humana se forman dos líneas, del centroide de la cabeza al del cuerpo, y del centroide del cuerpo al de las piernas. Las características de longitud y orientación extraídas de estas dos líneas son empleadas para detectar caída.
Se busca realizar dicho proceso con un bajo coste computacional.

2.3.3.2 Cambios de velocidad entre actividades normales y las caídas (mencionado en [2]).

Esta aproximación se basa en que, en actividades normales, la velocidad de cambio en el sujeto es menor que la velocidad que existe en eventos de caídas, la cual aumenta rápidamente.
Surgen problemas tales como ineficiencia en el sistema si el sujeto se acerca a la posición de la cámara, ya que aumenta su velocidad 2D.

2.3.4 Algoritmos 3D

Estos algoritmos están actualmente en pleno desarrollo. Son más complejos que los anteriores, por lo que implican un proceso de investigación mayor. Normalmente, se suele hacer uso de un sistema multicámara calibrado.
Entre los más conocidos tenemos:

2.3.4.1 Mediante un voxel person (mencionado en [8]).

Se trata de un método para reconocer las actividades humanas a través de un objeto en tres dimensiones que llamamos *voxel person*.
Este método utiliza una lógica difusa, donde la salida de cada nivel se suma alimentando al siguiente nivel. Si representamos un modelo basado en dos niveles para la detección de caída, tenemos que el primer nivel infiere los estados de la persona en cada imagen y el segundo trabaja en las sumarizaciones lingüísticas de los estados de la *voxel person*, realizándose la inferencia con respecto a la actividad. Las reglas que se usan para la detección de caída se suelen diseñar bajo la supervisión de personal cualificado y familiarizado con este tipo de eventos, como médicos o enfermeras, que saben con mayor certeza cómo las personas mayores suelen realizar sus actividades, para asegurar que el enfoque se está realizando adecuadamente.
Las ventajas residen en que es un enfoque bastante flexible, ya que pueden modificarse las reglas o ser eliminadas/añadidas otras nuevas con facilidad.
Lo que llamamos *voxel person* se obtiene de la siguiente manera: distintas cámaras posicionadas en la estancia capturan el vídeo desde diferentes puntos de vista, por lo que cada una va a extraer una silueta distinta en el proceso de segmentación. Se procede luego a reconstruir una única silueta mediante las distintas tomas que se han obtenido en cada cámara y se hacen interceder unas con otras para crear la *voxel person*.
Suponiendo, por ejemplo, que disponemos de dos cámaras, el esquema siguiente reproduce el procedimiento a seguir:

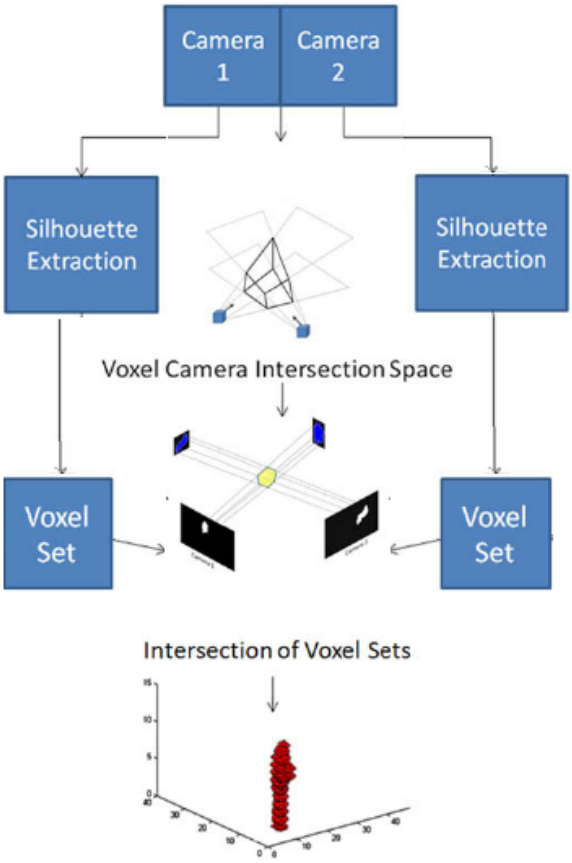

Figura 2-4: Reconstrucción de una voxel persona partir de dos cámaras independientes. Fuente: [8]

Con este tipo de sistemas se solucionan más fácilmente los problemas de oclusión provocados por los muebles de la estancia. Sin embargo, se necesita la calibración y sincronización de las distintas cámaras del sistema, lo que resulta caro y no trivial.

2.3.4.2 Mediante altura y área ocupada desde dos vistas ortogonales (mencionado en [9]).

Se propone un método que utiliza las medidas de las alturas y las áreas ocupadas por los sujetos en los que se desea detectar caída, diferenciando entre tres típicos estados: de pie o erguido, sentado y tumbado.

Esta estimación deriva del hecho de que la altura será mayor en el primero de estos tres estados que en los otros dos, así como el área ocupada será mayor estando en la posición de tumbado que en cualquiera de las otras dos. Se utilizan estos tres estados en combinación con dos vistas ortogonales procedentes de dos cámaras distintas, mediante las que se simplifica la estimación de área ocupada como el producto de las anchuras de la misma persona, vista desde cada una de las cámaras. Esto permite la estimación en tiempo real y además reduce en gran medida el coste computacional.

El evento de caída se detecta teniendo en cuenta el análisis que se realiza entre las transiciones de estados. Sin embargo, la estimación de características basada en el tamaño de las siluetas varía en toda la ventana de visualización debido a la perspectiva de la cámara. Para tratar con este problema, se propone utilizar Plantillas Empíricas Locales

(*Local Empirical Templates, LET*) que se definen como los tamaños de la postura "de pie" de las personas en imágenes locales. Éstas tienen dos importantes características: (1) es un método bastante automático y (2) mantiene la información de las perspectivas que se utiliza para la normalización de características de la imagen.

2.3.5 Algoritmos 2,5D

Esta aproximación es una solución intermedia entre los dos tipos de algoritmos descritos en los apartados anteriores. Se trata de algoritmos que siguen la filosofía de los implementados en dos dimensiones, 2D, pero que además incorporan la información de profundidad de la imagen, adquiriendo un enfoque similar al obtenido con algoritmos 3D y proporcionando muy buenos resultados.

Esta información de profundidad se puede obtener mediante diferentes métodos:

- Visión estéreo. Necesita dos cámaras que reconstruyan la secuencia, por lo que se hace necesaria la calibración y no es muy eficaz cuando la escena no es lo suficientemente texturada. Los algoritmos que llevan a cabo este método son caros y además no son útiles en condiciones de poca luminosidad.
- Time-of-Flight (ToF). Procura mayor precisión que los sistemas de visión estéreo, pero es muy caro y requiere imágenes de muy poca resolución (por ejemplo, 176x144).
- Luz estructurada. Es el principio que se lleva a cabo con un sensor o cámara Kinect [11][12][13], donde una luz estructurada de infrarrojos se proyecta en la escena y se observa con una cámara apta para ello. Estos sistemas pueden adquirir imágenes más grandes que con cámaras ToF a un precio mucho más bajo (por ejemplo, imágenes de 640x480 a 30 fps a un coste cinco veces menor que con cámaras ToF).

Como vemos la mayoría de enfoques basados en la información de profundidad adquieren dicha información mediante un sensor Kinect, ya que se considera un sistema *low-cost* y con una instalación sencilla que proporciona resultados eficientes. Algunos de estos enfoques son:

2.3.5.1 Mediante una cámara Kinect (mencionado en [11][12][13]).

Se trata de un método robusto contra la oclusión. Con el empleo de sensores Kinect se tienen dos ventajas principales:

- Se preserva la privacidad del individuo, ya que en la imagen no diferencia al sujeto de manera común, sino que está compuesta por una mezcla de colores en función de la profundidad de cada componente.
- Gracias a los rayos infrarrojos que utiliza este sensor, este sistema funciona tanto de día como de noche, es decir, tanto con escenas con luz como en ausencia de ella, una clara ventaja a tener en cuenta.

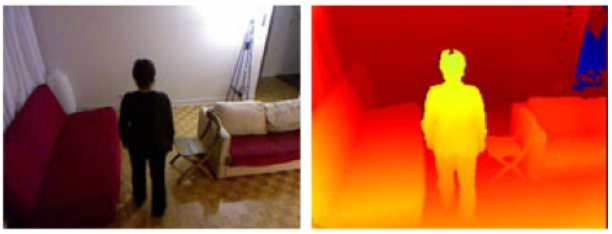

Figura 2-5: Escenas en condiciones normales de luminosidad. Fuente: [11]

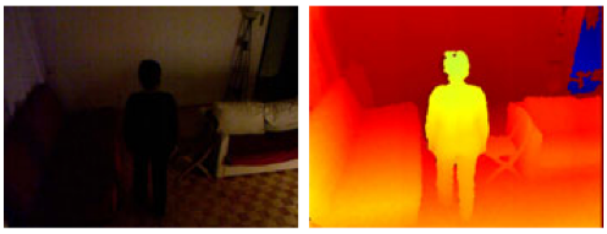

Figura 2-6: Escenas en condiciones de baja luminosidad. Fuente: [11]

Para este método de detección de caída seguido en [11] se siguen tres pasos principales:

1. Detección de plano de suelo, en lugar de una segmentación de fondo que suele ser el primer procedimiento a seguir en un procesado de vídeo convencional. Para esta detección del plano de suelo se hace uso de histogramas de valores de disparidad o desigualdad, técnica conocida como imágenes *V-disparity*. En la siguiente imagen podemos ver como se llega a la segmentación del plano de suelo mediante imágenes de profundidad en combinación con estos histogramas de disparidad:

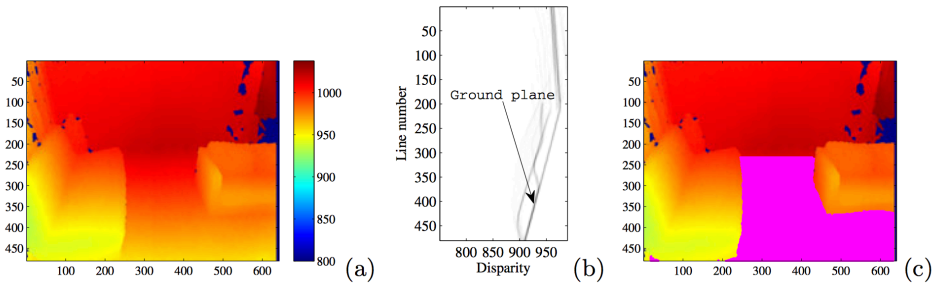

Figura 2-7: Segmentación de plano de suelo mediante V-disparity. Fuente: [11]

2. Localización y segmentación de la persona, para poder realizar su seguimiento.
3. Detección de caída, basándose principalmente en la distancia 3D del sujeto respecto al plano de suelo y la velocidad 3D del cuerpo.

Otros métodos que también emplean este tipo de sensores fusionan dos técnicas totalmente diferentes, como puede ser:

2.3.5.2 Combinación entre dispositivos portátiles y vídeo con cámaras Kinect (mencionado en [12]).

Mediante un acelerómetro se detecta una caída potencial, hipótesis que se verifica utilizando una imagen de profundidad obtenida como una toma de vídeo justo en el momento de dicha posible caída y, por último, se realiza una verificación final (para evitar falsas alarmas) con una imagen de energía, que expresa la distribución del movimiento de la persona en un conjunto de imágenes que preceden a la del momento de caída.

Se obtiene de este modo una reducción de falsos positivos y una alta tasa de detección efectiva. Además, se preserva en mayor medida la privacidad del sujeto por dos razones, la primera es que opera en condiciones de poca luminosidad y la segunda es que no obtiene tomas durante todo el rato que la cámara está habilitada, sino que sólo se activa cuando el acelerómetro ha detectado primeramente una caída potencial.

2.3.6 Conclusiones

Como se ha visto, existe un gran número de enfoques que persiguen crear algoritmos que detecten caídas, ya que es un ámbito que necesita de un amplio estudio e investigación por ser uno de los motivos más frecuentes de lesión en personas ancianas. Elegir uno u otro depende de muchos factores, tales como puede ser el coste computacional requerido o el presupuesto destinado para este fin.

2.4 *Datasets*

2.4.1 Introducción

A continuación mostramos los vídeos que han sido utilizados en el desarrollo de algunos de los algoritmos de las técnicas explicadas anteriormente. En algunos casos, estos *datasets* no son de dominio público.

En la mayoría de los casos los vídeos contienen más o menos la misma información: caídas simuladas, falsas caídas o falsos positivos (movimientos tales como agacharse o sentarse bruscamente) y actividades normales diarias. Incluyen también cambios de los objetos en la escena, diferentes tipos de ropa en colores y texturas, cambios de iluminación, oclusiones, etc.

2.4.2 *Dataset* de algoritmos 2D

En los métodos descritos en [2] no se proporcionan los videos con los que se realizan las pruebas.

Por su parte, en [6] los vídeos han sido creados especialmente para probar el algoritmo, son públicos y pueden descargarse en:

http://le2i.cnrs.fr/Fall-detection-Dataset?lang=fr

En [7], no se indicaba el acceso a los vídeos, pero se han pedido a sus autores para poder probar la eficiencia de su técnica y éstos han accedido a enviárnoslos.

2.4.3 *Dataset* de algoritmos 3D

En [8] se realiza la prueba del algoritmo implementado con los vídeos proporcionados en:

http://www.derektanderson.com/fallrecognition/datasets.html

En el trabajo desarrollado en [9], se utilizan los vídeos disponibles en

http://www.iro.umontreal.ca/~labimage/Dataset/

para comprobar los métodos generados.

2.4.4 *Dataset* de algoritmos 2,5D

De igual modo en [11], se utilizan los vídeos:

http://www.iro.umontreal.ca/~labimage/Dataset/

2.4.5 Conclusiones

Como conclusión, cada método posee sus propias ventajas y desventajas y dependerá de la aplicación y el objetivo de la misma, así como de los materiales disponibles, usar una técnica u otra. Por ejemplo, en el caso de 2,5D será necesario disponer previamente de una cámara Kinect.

2.5 *Conclusiones del estado del arte*

Una vez vistas las tres categorías en las que se ha clasificado las distintas técnicas existentes, se puede concluir que un sistema de video-monitorización es el óptimo para cualquier proceso de detección de caída, tanto por su eficacia como por la comodidad del sujeto, aunque existen ciertos temas algo controvertidos como el de la privacidad.

Sin embargo, se apuesta firmemente por esta técnica, de la cual existen infinitas posibilidades, varias de ellas recogidas aquí, y que depende en gran medida de los recursos disponibles y la necesidad de precisión rigurosa la elección de uno u otro de los sistemas, ya que, si bien un sistema 3D es el más eficaz y preciso, también es el que requiere una mayor complejidad y coste añadido, mientras que con un buen sistema 2D pueden llegar a conseguirse resultados bastante robustos y más asequibles para los usuarios.

3 Diseño y desarrollo del sistema

3.1 *Introducción*

Tras analizar el estado del arte y sopesar las posibilidades y alternativas posibles para la realización de un sistema de detección de caídas basado en el procesado de vídeo, se ha decidido tomar como referencia el trabajo desarrollado en [7], intentando reproducir cada una de las etapas diseñadas y posteriormente implementadas, tras cerciorarnos de disponer de los medios necesarios para poder reproducir este método.

Es por ello que ha sido necesario pedir expresamente los *datasets "Fall & Daily Activities Video Sample"* utilizados por sus autores, de manera que fuera posible ir comparando los resultados a la hora de asegurarnos de que el algoritmo seguía el funcionamiento esperado. Estos vídeos fueron adquiridos mediante una cámara IP no calibrada (Dlink DCS-920) con conexión Wi-Fi en formato MJPEG con una resolución de 320*240 píxeles.

Se trata de un método que detectará caídas en veinte vídeos en color. Estos vídeos consisten en una serie de secuencias que combinan actividades cotidianas: andar, sentarse o agacharse en cuclillas, así como movimientos y cambios de objetos; con una serie de caídas simuladas de diferentes maneras y posiciones: hacia delante, hacia detrás, de lado e incluso caídas como resultado de pérdida de equilibrio.
Además, se incorporan cambios de vestuario con diferentes colores y texturas, así como aquellas otras dificultades mencionadas en el estado del arte.

3.2 *Sistema original*

La figura humana es uno de los rasgos más simples que utilizan muchos algoritmos en la detección de caídas. Cuando una persona se cae, la figura humana cambia rápida y repentinamente, mientras que durante actividades normales o rutinarias cambia de forma más lenta o permanece constante en algunos aspectos. Este es el principio usado por la técnica que se propone en [7].
Esta técnica se basa en dicho movimiento de cambio tomando como referencia tres puntos concretos, los más representativos: cabeza, cuerpo y piernas. Una vez calculados estos tres puntos, se pueden derivar de ellos características como el cambio de orientación, sumatorios y razones de alturas, los cuales proporcionan gran información sobre el tipo de actividad realizada. La extracción de estos puntos es simple y poco complicada en comparación con otros de los métodos mencionados en el estado del arte.

En primer lugar, se realiza un método de segmentación de fondo basado en un filtrado de mediana, método de menor complejidad y coste computacional que produce unos resultados bastante asequibles en comparación con otros basados en mezcla de gaussianas, también utilizados para el fin que se persigue. El plano de fondo y la persona en

movimiento se detecta viendo la diferencia entre las *frames* entrantes con el modelo de fondo.

Una vez se ha detectado el primer plano o, de ahora en adelante, *foreground*, se busca caracterizarlo mediante tres puntos concretos, que serán los centroides de las tres diferentes regiones que se van a tener en cuenta. Primeramente, se calcula la *bounding box* de la persona que tenemos detectada en el *foreground* y se divide en tres porciones con una razón del 30%, 40% y 30% respecto del total (que serán, de manera aproximada, cabeza, cuerpo y piernas).
Estas medidas son una primera estimación, y llamaremos a las regiones R1, R2 Y R3. Con ellas, pasaremos a calcular, para cada una, tanto su altura como su anchura: $h_{R1,}$ h_{R2}, h_{R3} y $w_{R1,}$ $w_{R2\ y}$ w_{R3}, de la siguiente manera:

$$h_{Ri} = (0.4*i - 0.1*i^2)*H, \text{ si } H>W$$
$$h_{Ri} = H, \text{ en el resto de casos}$$

Ecuación 3-1

$$w_{Ri} = W, \text{ si } H>W$$
$$w_{Ri} = (0.4*i - 0.1*i^2)*W, \text{ en el resto de casos}$$

Ecuación 3-2

donde i=1, 2 y 3, correspondientes a los tres centroides, mientras que H y W son la altura y la anchura, respectivamente, totales de la *bounding box*. Una vez conocemos estos datos, podemos calcular las coordenadas de los centroides como:

$$g_{Rix} = 1/N_{Ri} * \Sigma\ (i=1,\ldots,\ N_{Ri})\ de\ x_1$$
$$g_{Riy} = 1/N_{Ri} * \Sigma\ (i=1,\ldots,\ N_{Ri})\ de\ y_1$$

Ecuación 3-3

donde N_{Ri} es el número de *foreground* píxeles, teniendo en cuenta . Los centroides serán (g_{R1x}, g_{R1y}), (g_{R2x}, g_{R2y}) y (g_{R3x}, g_{R3y}), para las regiones R1, R2 y R3, respectivamente.

El siguiente paso es considerar dos líneas, una desde P1 hasta P2 y la otra desde P2 a P3, como se muestra en la figura a continuación:

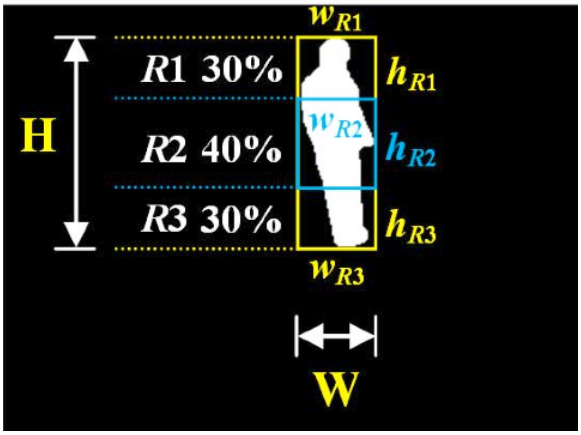

Figura 3-1: Técnica propuesta basada en el análisis de los parámetros extraídos. Fuente: [7]

Como cada línea representa la mitad de la porción del *foreground*, cualquier cambio en la distancia u orientación de la misma indicará que hay un cambio en la persona.
Obteniendo la distancia y la orientación de cada una de las dos líneas para cada *frame*, se lleva a cabo el análisis de cambio de la figura humana. Las distancias D1 y D2 entre puntos vienen dadas por:

$$D1 = ((g_{R1x} - g_{R2x})^2 + (g_{R1y} - g_{R2y})^2)^{1/2}$$
$$D2 = ((g_{R2x} - g_{R3x})^2 + (g_{R2y} - g_{R3y})^2)^{1/2}$$

Ecuación 3-4

Mientras que, para la información sobre la orientación, calculamos los ángulos que forman cada una de las líneas que tenemos calculadas (de P1 a P2 y de P2 a P3, respectivamente) con el eje horizontal x, de la siguiente manera:

$$\theta 1 = \arctan ((g_{R1y} - g_{R2y})/ ((g_{R1x} - g_{R2x}))$$
$$\theta 2 = \arctan ((g_{R2y} - g_{R3y})/ ((g_{R2x} - g_{R3x}))$$

Ecuación 3-5

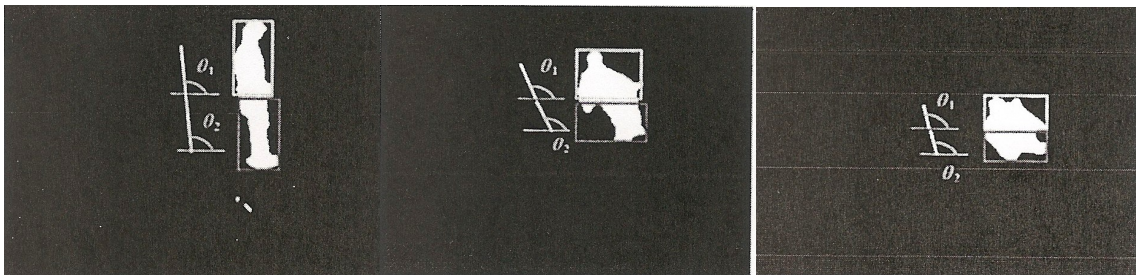

Figura 3-2: Distancias y orientaciones para: de pie, cayéndose y caído. Fuente: [7]

El enfoque se basa en el hecho de que diferentes cambios en la postura humana, por ejemplo de la posición de estar de pie a sentado o de estar de pie a caída, tendrán diferentes cambios entre la porción superior e inferior. Para ello, calculamos el ratio o la razón entre las dos distancias obtenidas y la diferencia entre los ángulos de ambas orientaciones, cuyo análisis servirá para la detección de caída.

La parte estrictamente de detección de caída se diseña como sigue:

Como en la mayoría de actividades normales la línea de la porción superior e inferior no cambiarán en gran medida (D1 y D2 serán muy similares entre ellas), el valor de la razón será igual a uno en la mayor parte de los casos. Es por ello que, cuando p cambie de un valor igual a uno a otro que sea distinto de uno vamos a considerar la posibilidad de que una caída se haya producido.

Por otro lado, en cuanto a la orientación, las porciones superior e inferior van a tener orientaciones similares en posiciones de estar de pie o estar tumbado. Por lo que, antes de la caída y justo después de la caída, la diferencia de orientación entre ambas porciones va a ser pequeña.

Partiendo de esta base, se procede a analizar la razón entre distancias de la *frame* anterior, p_{t-1} comparándola con la de la *frame* actual, p_t:

- Si no existe una posible caída, $\Delta p \approx 0$ ($p_t - p_{t-1} \approx 0$, ya que ambas serán aproximadamente valores cerca de 1) y las orientaciones son similares, θ1 se guarda como un ángulo de referencia, θ_r, y la suma de D1 y D2 se guarda como una longitud de referencia, D_r.

- Si existe una posible caída ($\Delta p>0$), buscamos las orientaciones θ1 y θ2 en la décima *frame* posterior a lo que consideramos como posible caída, y las llamamos θ_{N1} y θ_{N2}, respectivamente. Se ha considerado que en 10 *frames* ocurriría un movimiento rápido (como lo es una caída), pero se puede optar por otro valor. Se compara entonces:

$$\theta_{D1} = abs(\theta_{N1} - \theta_r)$$
$$\theta_{D2} = abs(\theta_{N2} - \theta_r)$$

Ecuación 3-6

  Esta diferencia debería ser muy pequeña, porque después de la caída la orientación es bastante similar, habiendo considerado un valor de menos de 10° para la diferencia en los ángulos.

Para superar el problema de mapear las dos porciones del cuerpo humano antes y después de la caída, se calcula la media de esas dos diferencias de orientación, μ_θ. Se considera caída cuando esta es mayor de 30°.
Sin embargo, existen caídas cuyos valores de μ_θ pueden ser iguales o menores que 30 grados. Por lo tanto, bajo esta condición, se comprueba la diferencia entre la longitud total de la línea en la décima *frame* tras la posible caída y la longitud de referencia D_r.
Basándose en las observaciones, la altura de la persona se verá drásticamente reducida durante una caída en el campo de visión de la cámara. El cambio en el sumatorio de las longitudes de las líneas después de una caída, D_{diff}, viene dado por:

$$D_{diff} = abs(D_r - (D_1 + D_2))$$

Ecuación 3-7

De este modo, determinan en [7] que, una caída es detectada como verdadera si D_{diff} es mayor que el 40% de la longitud de referencia, es decir si es mayor que $0.40*D_r$ píxeles. De lo contrario, se determinará que no se ha producido caída.

Por último y como comprobación final, una caída terminará con un período de inactividad si la persona está inmovilizada o inconsciente tras la caída. El último paso de la implementación propuesta es comprobar si existe algún movimiento de la persona después de la posible caída. La caída se confirma si la siguiente condición se cumple:

“El cambio en la distancia recorrida por el segundo centroide (correspondiente a la región del cuerpo) es menor o igual a 5 píxeles en 5 segundos, que se eligen como el período que se considera como inactividad, el cual podría modificarse, sobre todo aumentándolo, en caso de que se pretenda averiguar que la persona está totalmente inconsciente tras la caída.”

De aquí en adelante nos centramos en analizar cada uno de los pasos seguidos en profundidad.

3.3 *Diseño del sistema*

El diseño de nuestro sistema se ha basado fundamentalmente en lo expuesto en [7]. Si bien, algunas de las partes a reproducir no han sido totalmente fieles a éste, o por falta de profundidad en la información de la que se disponía o por considerar que otros enfoques podrían ofrecer mejores resultados.

En cuanto al modelado de fondo, necesario en cualquier procesado de vídeo, se ha realizado el filtrado de mediana que se mencionaba en la implementación original. Si bien, en todo el proceso restante no se daban detalles exactos, por lo que se ha implementado tras el estudio del estado del arte y la comparación de las distintas opciones, dados los medios disponibles y las características de los *datasets* con los que evaluar el sistema.

Concretamente, la caracterización del plano de *foreground* mediante los tres centroides que dividen distintas regiones del sujeto al que detectar caída se ha reproducido exactamente igual, aplicando las mismas fórmulas para calcularlos y extrayendo de igual modo los parámetros característicos, como distancia y orientación de las líneas que los unen.
Sin embargo, y aunque en general la implementación del bloque propio de detección de caída ha seguido el mismo esquema, se han introducido pequeñas modificaciones y alteraciones que nos proporcionan resultados más coherentes.

3.4 *Desarrollo*

3.4.1 Introducción

Tanto para el diseño como para el desarrollo del código de este algoritmo se ha procedido a dividir el trabajo en etapas o bloques, los cuales se han implementado de modo que, hasta que no se ha asegurado el comportamiento esperado en uno de ellos, no se ha avanzado al siguiente.

A grandes rasgos, la estructura que se ha seguido consta de los siguientes pasos: en primer lugar, se ha realizado un proceso de segmentación y modelado de fondo; a continuación se ha procedido a implementar cada una de las partes del algoritmo basado en el análisis de la variación de la figura humana y extrayendo aquellos parámetros que nos proporcionan la información deseada; finalmente, se ha estudiado cada movimiento calificándolo como caída o no caída, diferenciando además varios tipos en ésta.

Podemos generalizar que la implementación está basada en estos grandes bloques, que dividimos posteriormente en tres para comentarlos en profundidad:

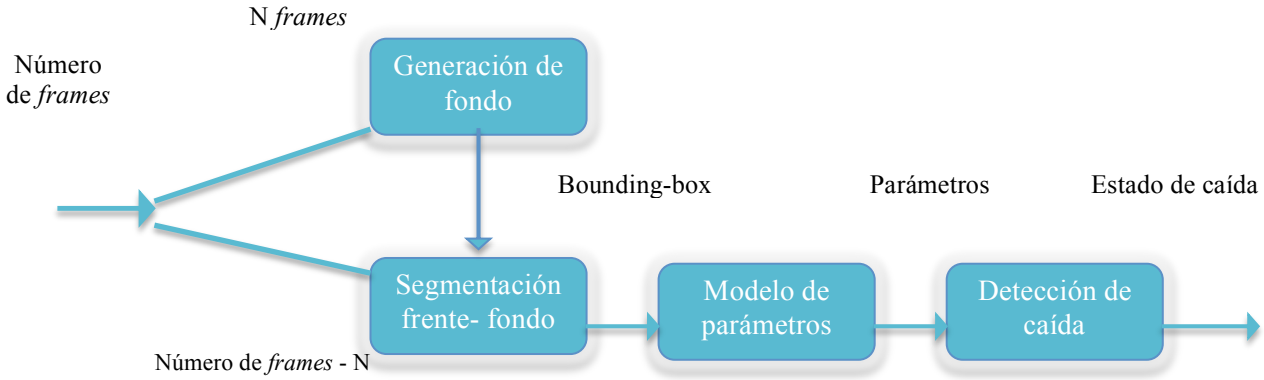

Figura 3-3: Diagrama de bloques del algoritmo implementado. Fuente: Propia.

- Generación del fondo y posterior segmentación frente-fondo.
- Extracción de parámetros característicos.
- Detección de sucesos: no caída o caída y, si la hay, de qué tipo.

Nuestro sistema ha sido implementado en la plataforma de trabajo ***Matlab*** al igual que en [7], los cuales indican las capacidades su ordenador: procesador Intel Core i3, a 2.13 GHz y memoria RAM de 4 GB, mientras que en esta implementación se ha utilizado un procesador Intel Core i5 a 2.4 GHz y memoria de 8 GB.
Es importante destacar que ha sido necesario realizar una descompresión del formato de los vídeos de los que disponíamos, imprescindible para que ***Matlab*** permitiera su lectura y su posterior descomposición en imágenes. Para esta descompresión del formato *.avi* se ha utilizado el programa ***Virtualdub***, el cual permite dicha tarea. Una vez leído se divide éste en sus distintas *frames*, las cuales se sitúan alrededor de las 550-800 para cada uno de los veinte vídeos y se procede a leer individual y consecutivamente cada una de ellas.
En cuanto a la organización del programa implementado tenemos que la mayoría del código principal se estructura en un único archivo *fall_detection.m*, donde se realizan los tres bloques que se detallarán a continuación. Sin embargo, se crean también algunas funciones que serán llamadas desde dicho programa principal. Estas funciones, todas de implementación propia, son:

- *umbralÓptimo = UmbralizaGlobalOtsu(ima)*
- *ima_bin = binarización(ima, umbral1, umbral2, umbral3)*
- *caída = check_inactivity(centroid1, centroid2)*
- Y, adicionalmente, cada vez que se quiera utilizar un nuevo vídeo, habrá que utilizar primero un pequeño programa creado, *readandsave,* que descompondrá el vídeo en distintas *frames* para después leerlas desde el código principal, aunque

también se requerirá el vídeo completo para averiguar el número de imágenes correspondientes a cada vídeo en el programa principal.

3.4.2 Proceso de generación de fondo y segmentación frente-fondo.

3.4.2.1 Obtención del background y la imagen diferencia.

En concreto, para esta etapa no se proporciona información detallada en [7] sobre su diseño, dado que se entiende que es un preprocesado fundamental para cualquier algoritmo basado en vídeo. Por esta razón, este bloque ha sido sin duda el que ha requerido más tiempo y esfuerzo, ya que constituye la etapa o una de las etapas más importantes que determinará en gran medida el comportamiento y la precisión del sistema.

Existen diferentes técnicas de segmentación de fondo, como primer paso para la detección del objeto de interés en la escena.
En este caso, se podría generalizar diciendo que sólo nos es relevante aquello que está en movimiento durante la secuencia de vídeo, la persona, teniendo en cuenta que pueden darse algunas excepciones como los cambios de posición de objetos. Por tanto, el objetivo de este primer bloque es extraer al sujeto o persona en el vídeo de entrada, de modo que así podamos analizar los rasgos y características que nos ayuden a detectar la caída.

Dentro del gran rango de algoritmos que han sido diseñados para este fin, uno de los más simples pero eficientes métodos expuestos en [3][7] es el modelado de fondo mediante la imagen diferencia entre cada una de las *frames* entrantes y una *frame* reconocida como "sólo fondo". Esta imagen "sólo fondo" puede obtenerse fácilmente mediante un filtro de mediana en aquellas *frames* en las que aún no ha aparecido la persona a detectar. Este filtro de mediana debe aplicarse para cada canal o plano de color por separado.
Para nuestro algoritmo y habiendo previamente observado los vídeos de prueba, se ha establecido un valor orientativo Tfondo = 10 *frames* en las que realizar dicho filtrado. En algunos de los vídeos, el número de *frames* de "sólo fondo" es aún mayor que Tfondo, pero en cambio, para otros este valor es una buena aproximación.

Una vez tenemos identificado el fondo, que de ahora en adelante denominaremos *background,* podemos calcular la imagen diferencia que se va obteniendo al leer las siguientes e ir restándolas consecutivamente con el *background* estimado.

3.4.2.2 Umbralización.

Ya obtenida la imagen diferencia es necesario umbralizarla, de modo que quede una imagen totalmente bimodal donde se distingan con total claridad el *background* de la persona a la cual queremos detectar la caída.

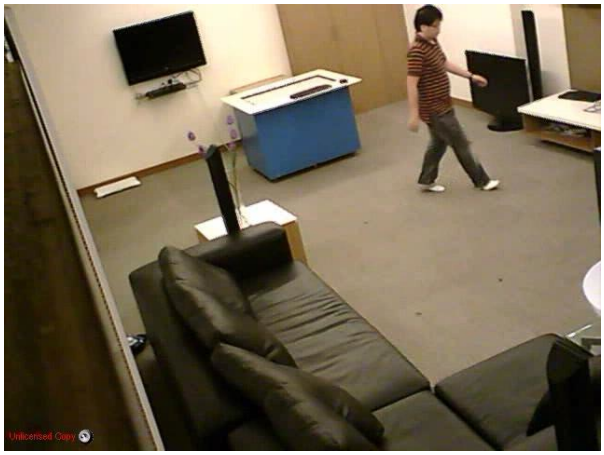

Figura 3-4: *Frame* del vídeo original previa a la umbralización. Fuente: [7]

Figura 3-5: Correspondiente *frame* umbralizada mediante Otsu. Fuente: Propia.

Para realizar dicha umbralización se ha decidido utilizar el método de Otsu, el cual es uno de los más extendidos por sus buenos resultados, rapidez y eficacia computacional. Esta umbralización ha de hacerse, de igual modo a como hacíamos para el filtrado de mediana en el *background*, para cada plano de color por separado. Posteriormente, se llevaría a cabo la binarización de la imagen teniendo en cuenta estos tres umbrales (uno para cada plano de color R,G y B) y estableciendo la condición necesaria para que éstos converjan en uno sólo. Esto es necesario porque, de lo contrario, se crearían colores falsos en la imagen como consecuencia de haber establecido los umbrales por separado para cada canal.

La umbralización de la imagen diferencia debe de hacerse sólo y exclusivamente cuando dicha imagen no sea nula, es decir, cuando aparezca la persona en la imagen. Si no fuera así, el método de Otsu intentaría umbralizar el ruido de la imagen, obteniéndose un resultado absolutamente no deseado.

Un caso especial y que es importante destacar es que en algunos de los vídeos el valor de Tfondo podría ser mayor, es decir, la persona no aparecía exactamente en la *frame* Tfondo+1 para todos ellos.
Por esta razón, se ha tenido en cuenta este hecho agregando una condición en la que, si la imagen diferencia no es totalmente nula (para ello previamente se redondea la imagen a su

valor entero inferior más próximo) se iguala la nueva imagen binaria con esta imagen diferencia, que será completamente negra o, lo que es lo mismo, nula. Esto ocurrirá en aquellas *frames* mayores que Tfondo, pero en las que no aparece aún la persona.

3.4.2.3 Aplicación de operadores morfológicos y obtención de la bounding-box.

La siguiente etapa se basa en aplicar operadores morfológicos a la imagen binaria. Esto se realiza como método para conseguir un mejor grado de definición de la silueta humana, sin demasiados huecos y cortes en su interior.

Para ello se decide hacer una operación de cierre, lo cual se basa en una dilatación, que cerrará los huecos que existan en el interior de la silueta debido a la propia imagen, como por ejemplo ocurre por las rayas de la camiseta que lleva puesta la persona que aparece en la escena; seguida de una erosión, que volverá a disminuir el tamaño que había sido extendido por la dilatación, pero con la ventaja de que los huecos, que ya han sido rellenados, no cambiarán.

Este tipo de operaciones morfológicas necesitan un elemento estructurante, es decir, un elemento de forma y tamaño determinados en función de la geometría de la imagen sobre la que se pretende aplicar la operación. Normalmente, una operación de cierre estricta requiere de un mismo elemento estructurante tanto para la dilatación como para la erosión. Sin embargo, en este caso y tras múltiples pruebas con distintos elementos, se ha optado por combinar un elemento cuadrado de tamaño igual a 8 píxeles para la dilatación y otro cuadrado pero de 4 píxeles para la erosión. De este modo, conseguimos que se rellenen los huecos en la medida de lo posible pero que la imagen no pierda demasiados bordes a la hora de volverla a erosionar.

El siguiente paso es crear la *bounding-box*, uno de los métodos más comúnmente usados, como se especifica en [2][3][4][6][7] . Ésta se define como la razón entre la altura y la anchura del objeto "encerrado".

En Matlab existe una función predefinida que calcula la *bounding-box* de manera automática, por lo que la utilizamos en nuestra implementación:

```
regionprops(L, 'BoundingBox')
```

siendo L una matriz obtenida con la función de Matlab *bwlabel(),* que se utiliza para el etiquetado de una imagen binaria, especificando la conectividad de sus elementos. Esto es, se creará una matriz que tendrá todos los píxeles etiquetados como fondo o *background* a 0, todos los píxeles conectados que conformen un mismo objeto tendrán un valor de 1, los de otro objeto diferente valdrán 2, y así sucesivamente. De este modo, se creará una *bounding-box* específica para cada elemento etiquetado con un valor distinto de la imagen (una para el elemento de valor 1, otra para el de valor 2, etc.).

Por esta razón, una vez se han hecho estos pasos es necesario juntar todas las *bounding-boxes* creadas en una sola, la cual corresponderá a la figura de la persona en su totalidad. Esto es importante y necesario ya que, aunque se han aplicado operaciones morfológicas sobre la imagen, no siempre se consigue unificar toda la silueta en un mismo bloque, por lo

que si algunos píxeles separan dos regiones que deberían estar unidas, al crear una única *bounding-box* quedan unificadas. Esto es posible gracias a que la función utilizada devuelve, para cada *bounding-box* los siguientes parámetros:

[x, y, anchura, altura]

por lo que, conociendo éstos y estableciendo las condiciones adecuadas, se consigue el objetivo buscado.

Figura 3-6: Correspondiente *frame* tras la aplicación de operadores morfológicos. Fuente: Propia.

3.4.3 Extracción de parámetros característicos

A esta altura ya podemos extraer los parámetros que van a caracterizar a la persona y a determinar su estado de caída. Es decir, ya es posible realizar la representación basada en los tres puntos descritos: cabeza, cuerpo y piernas. Esto puede hacerse a partir de la imagen binaria y los datos obtenidos de la *bounding-box* de manera que, conociendo su altura y anchura, así como sus coordenadas x e y, se establecen las regiones R1, R2 y R3, que corresponden al 30, 40 y 30%, respectivamente, de la persona total, como se veía en la Figura 3-1: Técnica propuesta basada en el análisis de los parámetros extraídos. Fuente: [7]

A continuación se procede al cálculo de los centroides, uno para cada región descrita:

- En primer lugar, es necesario obtener el número de píxeles de foreground y el número de píxeles totales, tanto de *foreground* como de *background*, de nuevo para cada región. Esto puede hacerse de manera sencilla en ambos casos, siendo un poco más laborioso para el segundo de ellos pero, dado que tenemos los píxeles de *foregound* binarizados con valor 255, podemos realizar un sumatorio que acumule todos los píxeles con este valor para cada región.
- Una vez se tienen estos valores, se van guardando en dos vectores las coordenadas x e y, respectivamente, de los píxeles de *foreground* con valor 255. Después, es necesario reajustar su posición para que quede con respecto a la de la *bounding-box*, ya que en primera instancia se obtiene respecto a la imagen binaria total.

Finalmente se suman todos los valores obtenidos y se normaliza con el valor calculado de píxeles de *foreground*.

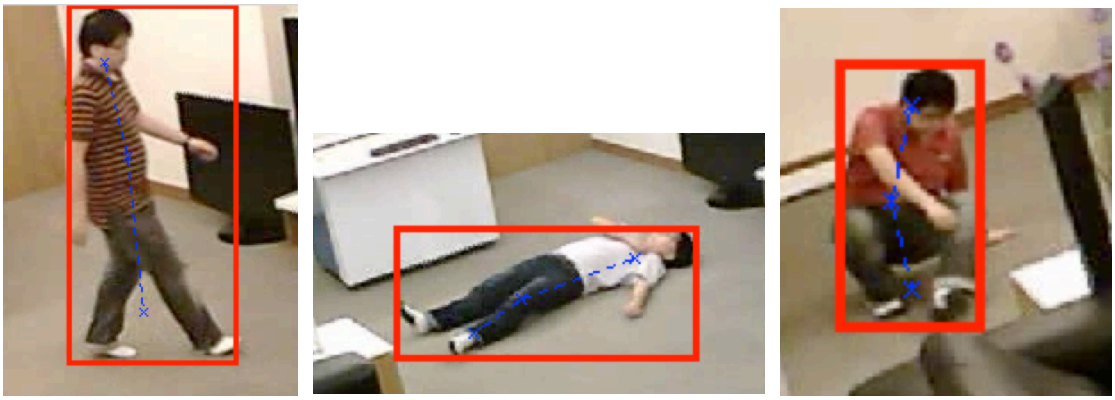

Figura 3-7: Representación de parámetros característicos: centroides y líneas, para distintos estados de caída sobre las *frames* originales. Fuente: Propia.

Obtenidos ya los tres centroides, es fácil crear las dos líneas que los une y calcular su longitud o distancia que separa a cada par de ellos, así como la orientación o el ángulo que forman con la horizontal. Analizando los valores que toman estos parámetros se puede detectar el estado de caída.

3.4.4 Detección de caída

En esta sección se explica en profundidad cada uno de los pasos seguidos para conseguir detectar caídas. Para ello, en primer lugar es necesario disponer de ciertas variables y/o parámetros que se utilizarán en la detección. Estos son:

- *frame_total*: Se trata de un *cell* que contiene en cada una de sus celdas 5 variables: la *frame* correspondiente y sus cuatro parámetros característicos D1, D2, θ1, θ2. Hay que tener en cuenta que, en aquellas *frames* en las que no se detecta persona, estos parámetros serán nulos, y en las que se utilizan exclusivamente para el modelado de fondo, las N primeras *frames*, estará vacío.

- *centroid*: Otro *cell* que guarda la *frame* correspondiente y sus tres centroides con coordenadas x e y calculados en la extracción de parámetros. Este *cell* estará vacío en aquellas *frames* en las que no se detecta persona.

Por otro lado, se creará un nuevo *cell* que almacenará los distintos estados de caída que se detecten durante el proceso. Estos estados serán:

Estado de caída	Etiqueta
-2	No analizado
-1	No persona; inicialización
0	Persona en estado normal; no caída
1	Falsa alarma
2	Alarma; cayéndose
3	Caído y levantado
4	Caído e inmóvil

Tabla 3-1: Valores dados en el análisis del algoritmo a distintos estados de caída. Fuente: Propia.

Para facilitar la lectura y comprensión de la implementación de este bloque, se anexa un pseudocódigo al final de la memoria, Anexo A. En él, utilizamos un bucle para analizar todas las *frames* de la secuencia. Sin embargo, es necesario establecer un par de condiciones que aseguren el funcionamiento correcto del programa. Por ello, hay que contemplar aquellos casos en los que *frame_total* está vacío, o bien, es nulo. Concretamente, debe tenerse en cuenta que este requisito debe cumplirlo tanto la *frame* actual, como la anterior, como la frame correspondiente al *Tsalto* posterior, por ser necesarias éstas y sus parámetros extraídos en el paso anterior para establecer las distintas comparaciones que determinen el estado de caída.

Una vez se han cumplido este tipo de requisitos y condiciones, se pasa a calcular, mediante las variables guardadas en *frame_total*, los distintos parámetros característicos en la *frame* actual y la inmediatamente anterior, es decir en t y t-1: D1, D2, θ1, θ2. Con estos, ya es posible calcular las razones p_t y p_{t-1} para posteriormente calcular el incremento, Δp, producido entre ambas.
Cuando Δp supere un cierto valor pequeño ε, que se ha fijado experimentalmente a 0.2, se considera que existe un posible cambio en la persona analizada y que, por tanto, una caída ha podido producirse. Por ello, se comprueba esta condición, estableciendo por defecto no caída (0) siempre que no la cumpla y calculando, dado este caso, una longitud de referencia como la suma de las dos distancias D1 y D2 de la *frame* actual y una orientación de referencia como la orientación θ_1 de la *frame* actual (D_r y θ_r).

Una vez se cumple la condición, se pasa a comprobar el estado de la caída de la *frame* que se analiza. Para llevar a cabo esta tarea acudimos a un cierto número de *frames* posteriores a la actual determinado por *Tsalto*, que se ha establecido, siguiendo lo expuesto en [7], en un valor de 10. En este punto, se calculan las orientaciones para esta décima *frame* posterior y se realiza una comparación al restarlas con las guardadas anteriormente como referencia. Acto seguido se obtiene la media entre ambos resultados, μ_θ. Si esta constante supera un valor de 30°, según el sistema original, se establece una alarma de caída (2). En caso contrario, dado que pueden existir caídas que no cumplan esta condición estrictamente, se calcula una diferencia, D_{diff}, entre la longitud total de la línea en la décima

frame tras la posible caída y la longitud de referencia D_r. Si este valor es mayor que el 40% de la longitud de referencia se determina que se ha producido una caída (2) y, si no, se cataloga como una falsa alarma.

Cuando se obtiene una potencial caída (2) ha de corroborarse acudiendo a la función *check_inactivity()*, la cual comprobará si el sujeto se ha caído pero ha vuelto a levantarse (3) o si, transcurrido un cierto tiempo, permanece aún en el suelo inmóvil (4).

A efectos prácticos, este último caso sería de mayor gravedad que el primero, ya que la persona puede haber caído inconsciente, por lo que la alarma debe ser mayor que en el otro caso, en que ha conseguido levantarse.

Esta función recibe dos parámetros: el *cell* que contiene los centroides en la *frame* actual y en la *frame* correspondiente al salto *Ttipocaida* posterior. Según lo establecido en [7], esta constante corresponde a un valor igual a 150 frames, ya que los vídeos han sido grabados a 30 fps y el período de inactividad establecido en una primera aproximación es de 5 segundos. Dentro de la función se calculará la distancia euclídea existente entre los dos centroides correspondientes a la región intermedia, el cuerpo. Se establecerá el estado de caída (3) si ésta cambia durante el tiempo establecido más de 5 píxeles, y en caso contrario se determinará el estado (4). Establecido este estado, todas las *frames* que pertenecen al salto dado y que no han sido analizadas se ponen a (-2).

Es importante hacer referencia a una serie de casos concretos en los que no podrá llamarse a dicha función. Esto ocurre cuando se intenta comprobar el estado de caída acudiendo a la 150 *frame* posterior y ésta, o bien es nula, o bien se sale del rango de *frames* disponibles. Por ello, cuando se da este caso, se avanza t hasta situarlo en la última *frame* en rango, de modo que en la siguiente iteración no siga calculando parámetros, ya que no podrá corroborarse la caída llamando a *check_inactivity()*.

Otra consideración que debe hacerse es que, el centroide de la 150 *frame* posterior sea nulo, debido a que corresponda a una *frame* en la que ya ha desaparecido la persona. En dicho caso, tampoco podrá corroborarse la caída.

A continuación, se muestran varios esquemas típicos que arrojaría el algoritmo según los valores dados en su implementación:

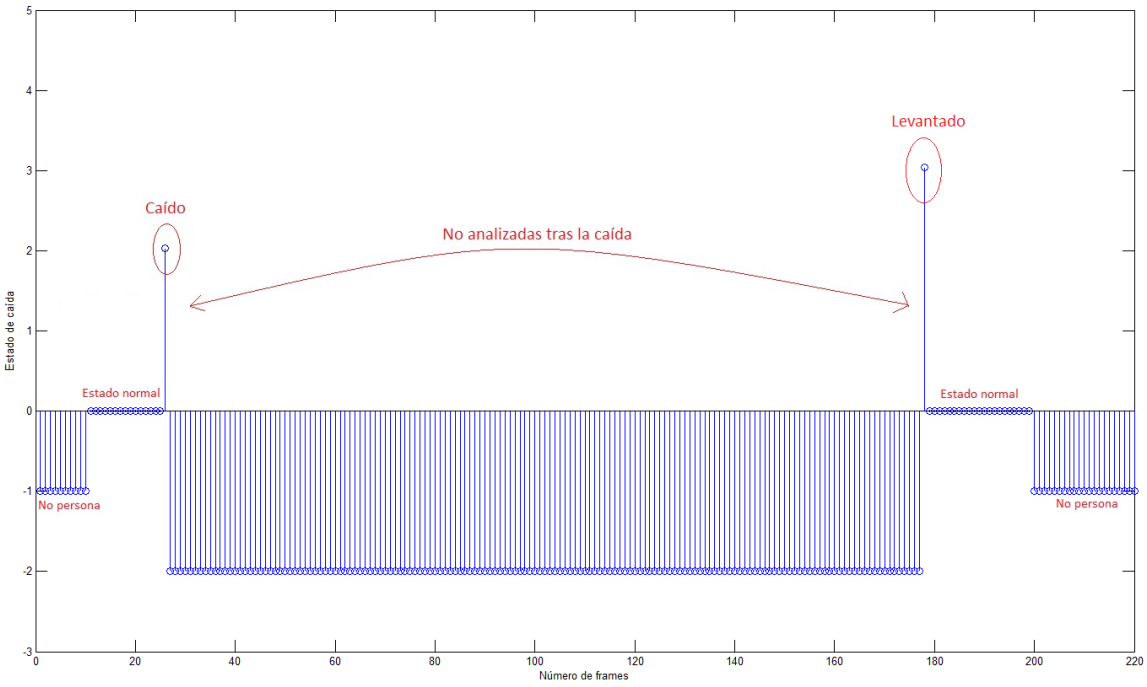

Figura 3-8: Esquema de estados normal + caído + levantado. Fuente: Propia

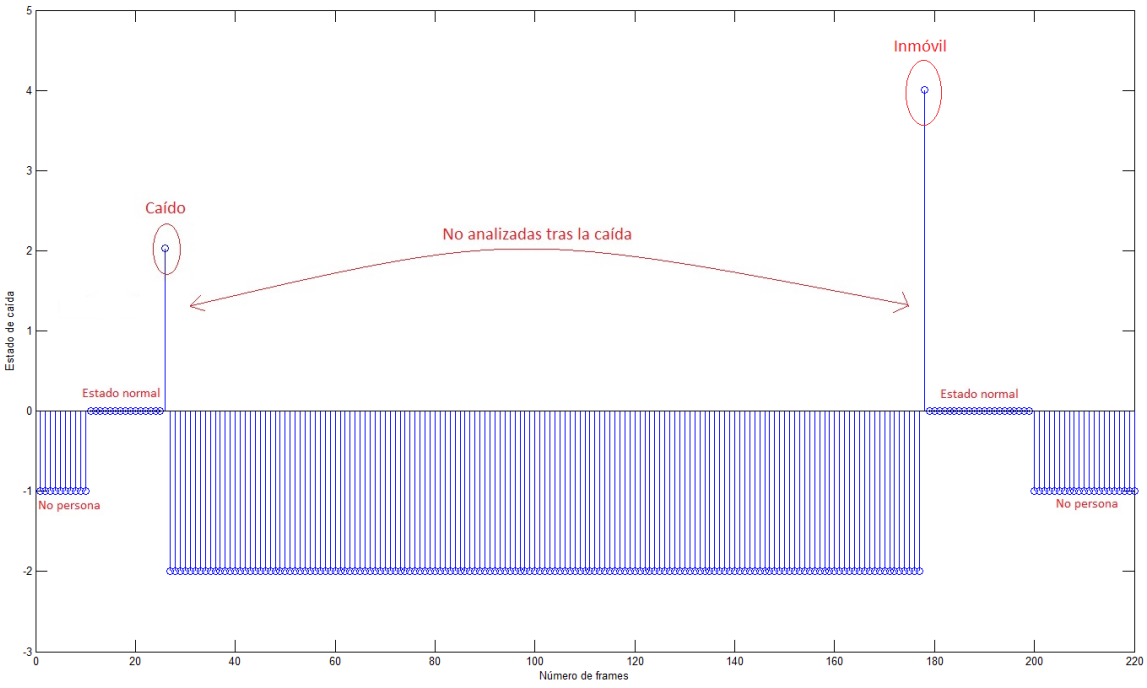

Figura 3-9: Esquema de estados normal + caído + inmóvil. Fuente: Propia

4 Evaluación del sistema

4.1 *Introducción*

En este apartado se van a estudiar y a analizar los resultados arrojados por el algoritmo implementado, a través de una serie de pruebas con distintos vídeos de características variadas. Se proporcionarán de igual modo los resultados obtenidos tras estas pruebas, obteniendo la calidad del sistema generado.

Para ello, se utilizará el primero de los sistemas vistos en el estado del arte, basado en algoritmos en dos dimensiones y al cual está orientada la implementación. De este modo, se pretenderá detectar caídas con la mayor precisión y fiabilidad posible, estableciendo un método robusto para el fin que se persigue.

4.2 *Pruebas*

A continuación, se realizan las pruebas con el *dataset* disponible, formado por un conjunto de veinte vídeos, proporcionados directamente por los autores de la implementación original [7].
Estos vídeos contienen:

- Actividades cotidianas tales como estar de pie, caminando.
- Actividades que pueden ser confundidas por el algoritmo con una caída, tales como sentarse bruscamente, agacharse, ponerse en cuclillas.
- Caídas simuladas, realizadas desde distintos ángulos y posiciones.
- Cambios de vestuario con diferentes colores, estampados y texturas.
- Entradas y salidas del campo de visión de la cámara.
- Oclusiones de objetos y de la persona por objetos, así como cambios en la posición de los muebles.
- Sombras y cambios de iluminación.

4.2.1 *Ground truth*.

El *dataset* que se nos proporciona no lleva un *ground-truth*[3] asociado, por lo que para llevar a cabo la evaluación del sistema, se ha optado por crear un *ground truth* para cada uno de los vídeos a analizar. Estos archivos se han construyen anotando cada una de las *frames* del vídeo correspondiente junto con su estado de caída observado experimentalmente. De tal manera que, para cada *ground truth* se adjudican los siguientes valores:

[3] Conjunto de datos objetivos para comprobar la exactitud o precisión del conjunto de entrenamiento disponible.

Estado de caída	Etiqueta
-1	No persona
0	Persona en estado normal; no caída
2	Alarma; cayéndose
3	Caído
4	Levantándose

Tabla 4-1: Valores dados en el *ground truth* a los distintos estados de caída. Fuente: Propia.

4.2.2 Matching

Como se puede observar en estos valores, a diferencia de aquellos que establecía el propio algoritmo, ha desaparecido el valor (1), que indica falsa alarma, ya que a simple vista no se aprecia aquello que el algoritmo puede calificar de tal modo.
De igual modo, las *frames* que el algoritmo calificaba como (-2), correspondientes a aquellas que no se analizaban por encontrarse en el salto dado tras detectar la caída y acudir a check_inactivity a corroborar y definir su tipo, aquí tampoco se anotan con ese valor, sino que cierta parte corresponderán a un (2), cayéndose, otra parte a (3), caído y otras tantas a (4), levantándose.
Por último, en las *frames* que en el *ground truth* se anotan como (4), es decir, que la persona caída ya se ha levantado, en realidad corresponden con un (0), no caída y estado normal, en los resultados arrojados por el algoritmo. Se recuerda que el algoritmo, cuando detectaba una caída, acudía a la 150 *frame* posterior y determinaba si el sujeto aún permanecía en el suelo inmóvil o, si por el contrario, se había vuelto a levantar. Esto en el *ground truth* no es así, por lo que sólo coincidirá el valor de la *frame* 150 posterior exactamente, y todas aquellas posteriores que en el *ground truth* se irán calificando como (4), por estar el sujeto levantándose, nuestro sistema lo establece como un (0), estado normal sin caída.

Es por esta razón por la que es necesario realizar un *matching* o hacer coincidir aquellos valores que difieren entre el estado de caída observado al crear el *ground truth* y aquel determinado por el algoritmo pero que, en realidad, corresponden a una misma acción.
Tras realizar este *matching*, se procede a comparar ambos resultados, dibujando un gráfico que muestre, en cada caso, los valores tanto para el *ground truth*, como para los del propio estado de caída determinados por el algoritmo implementado.

4.2.3 Scores

En función de las distintas combinaciones posibles entre los resultados arrojados por el algoritmo y los asignados en el *ground truth* se realizarán las siguientes clasificaciones:

- ***True Positive***: Se detecta TP cuando el sistema detecta caída y acierta.
- ***True Negative***: Se detecta TN cuando el sistema detecta no caída y acierta.
- ***False Positive***: Se detecta FP cuando el sistema detecta caída y se equivoca.
- ***False Negative***: Se detecta FN cuando el sistema detecta no caída y se equivoca.

	actual positive	actual negative
predicted positive	TP	FP
predicted negative	FN	TN

Tabla 4-2: Scores utilizadas para evaluar el algoritmo. Fuente: [14]

Para entender la tabla que se muestra a continuación es necesario establecer una serie de condiciones, ya que estrictamente sólo tenemos dos estados: caída y no caída, en función de los cuales se realizará la asignación de uno de los cuatro tipos de *score*. Por ello, se señala que:

- → Para el *ground truth*: los estados: no persona y estado normal se toman como "no caída"; mientras que los tres restantes: cayéndose, caído y levantándose, se tienen en cuenta como caída, ya que en cualquier de ellos ha ocurrido una caída.
- → Para el análisis: los estados: no analizado, no persona, estado normal y falsa alarma se toman como "no caída", y los restantes: caída, caído+levantado y caído+inmóvil;, se toman como "caída", por la misma razón mencionada anteriormente.

Por otra parte, se indica también que en el estado "no persona" se tienen en cuenta tantos las *frames* dedicadas al *background* en la segmentación de fondo como aquellas entrantes tras esta segmentación en las que aún no ha aparecido la persona.

Por último, comentar que las *frames* correspondientes a "no analizado" son aquellas que quedan en el salto dado tras detectar caída y acudir a la función *check_inactivity()*.

Análisis / Gt	No analizado	No persona	Estado normal	Falsa alarma	Caído	Caído + levantado	Caído + inmóvil
No persona	TN	TN	TN	TN	FP	FP	FP
Estado normal	TN	TN	TN	TN	FP	FP	FP
Cayéndose	FN	FN	FN	FN	TP	TP	TP
Caído	FN	FN	FN	FN	TP	TP	TP
Levantándose	FN	FN	FN	FN	TP	TP	TP

Tabla 4-3: Distintas combinaciones posibles de *scores* entre el análisis y el *ground truth*. Fuente: Propia

4.2.4 Métrica

Una vez hemos obtenido estas puntuaciones podemos calcular la eficiencia y el rendimiento de nuestro sistema aplicando las siguientes fórmulas [14] comúnmente utilizadas para llevar a cabo esta tarea:

$$Precision = \frac{TP}{TP + FP}$$ **Ecuación 4-1**

$$Recall = \frac{TP}{TP + FN}$$ **Ecuación 4-2**

En concreto, el valor de *precision* aumenta cuando hay pocos falsos positivos. Mide que las instancias clasificadas como clase "caída" sean realmente de la clase "caída", aunque haya instancias de dicha clase que se clasifiquen como otra diferente.
Por otro lado, el valor de *recall* aumenta cuando hay pocos falsos negativos. Mide que las instancias de la clase "caída" se clasifiquen como clase "caída", aunque otras instancias también se clasifiquen como tal sin serlo.

Otras medidas ampliamente utilizadas que aportan más información adicional sobre el sistema son:

$$True\ negative\ rate = \frac{TN}{TN + FP}$$ **Ecuación 4-3**

$$False\ positive\ rate = \frac{FP}{FP + TN}$$ **Ecuación 4-4**

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$ **Ecuación 4-5**

Todos estos valores se encuentran entre 0 y 1.

4.3 *Resultados*

4.3.1 Resultados de la métrica

Siguiendo con los resultados, en la siguiente tabla se muestra el rendimiento del algoritmo implementado para la mayoría de los veinte vídeos de prueba, incluyendo los casos más representativos y particulares.

Es importante destacar que se ha decidido establecer en algunos casos un NA (No Aplica). Por ejemplo, esto ocurre cuando en el sistema se utilizan para las pruebas *datasets* en los que no ocurre ninguna caída (**vídeos 10, 13, 14, 16, 17**) y, correctamente, no se encuentra ningún TP (ninguna caída detectada como caída) ni ningún FP (ninguna caída no detectada), por lo que *precision* sería 0. Esto ocurre no porque el sistema no funcione del modo esperado, sino porque no procede analizar ese caso ya que no existen valores que introducir en la fórmula.

Siendo así, se van a representar los resultados para dos casos diferentes:

- El primero de ellos, siguiendo lo explicado en el *matching*. Esto quiere decir que las *frames* establecidas con un estado (-2) se hacen corresponder con aquellas que el *ground truth* calificaba como 3, es decir, cuando el sujeto está caído en el suelo, ya que idealmente, cuando el sistema detecte caída y acierte, así será. Esta opción da lugar a "sobrepuntuaciones", es decir, en caso de que el sistema falle, todas las *frames* no analizadas que quedan en el salto se van a tomar como caída y puede coincidir que entre ellas, alguna caída haya ocurrido sin ser detectada, dando puntuaciones correctas cuando en realidad el sistema había detectado una caída donde no debía. En este caso tenemos los siguientes resultados:

	TP	**TN**	**FP**	**FN**	***Precision***	***Recall (True positive rate)***	***True negative rate***	***False positive rate***	***Accuracy***
Vídeo 1	190	244	2	131	0.9895	0.5919	0.9919	0.0081	0.7654
Vídeo 2	187	270	1	99	0.9941	0.6538	0.9963	0.0036	0.8205
Vídeo 3	274	302	1	115	0.9964	0.7044	0.9967	0.0033	0.8324
Vídeo 6	399	341	3	20	0.9925	0.9523	0.9913	0.0087	0.9699
Vídeo 7	140	389	1	230	0.9929	0.3784	0.9974	0.0025	0.6961
Vídeo 9	154	289	3	24	0.9809	0.8652	0.9897	0.0102	0.9426
Vídeo 10	0	508	5	0	NA	NA	0.9903	0.0097	0.9903
Vídeo 11	83	216	4	79	0.9540	0.5123	0.9818	0.018	0.7827
Vídeo 13	0	636	5	0	NA	NA	0.9922	0.0078	0.9922
Vídeo 14	0	260	2	0	NA	NA	0.9924	0.0076	0.9924
Vídeo 16	0	450	5	0	NA	NA	0.9890	0.0109	0.9890
Vídeo 17	0	688	8	0	NA	NA	0.9885	0.0116	0.9885
Vídeo 18	145	249	1	103	0.9932	0.5847	0.9960	0.0040	0.7912
Valores medios	**196.5**	**287.5**	**21.25**	**100.125**	**0.9866**	**0.6553**	**0.9926**	**0.0073**	**0.8251**

Tabla 4-4: Resultados obtenidos en la evaluación del sistema con los *datasets* disponibles: primera opción. Fuente: Propia

- Por otro lado, podemos optar por una solución más precisa y que no "sobrepuntúe". Esto es, cada vez que el sistema detecte *frames* con estado (-2), se las salta y no las analiza. Del tal modo, se evitará que se tomen como aciertos aquellos que no lo son, sin afectar al rendimiento del sistema en caso de que se detecte caídas correctamente. Si el sistema fuera riguroso y funcionara óptimamente, la proporción entre ambas comparaciones sería mínima y los resultados serían más o menos los mismos. Para esta segunda opción, tenemos:

	TP	TN	FP	FN	*Precision*	*Recall (True positive rate)*	*True negative rate*	*False positive rate*	*Accuracy*
Vídeo 1	3	103	2	131	0.6000	0.0224	0.9852	0.0148	0.5056
Vídeo 2	4	155	1	99	0.8000	0.0388	0.9936	0.0064	0.6139
Vídeo 3	6	123	1	115	0.8571	0.0496	0.9919	0.0081	0.5265
Vídeo 6	5	139	3	20	0.6250	0.2000	0.9789	0.0211	0.8623
Vídeo 7	6	76	1	230	0.9929	0.0254	0.9870	0.0130	0.2620
Vídeo 9	2	143	3	24	0.4000	0.0769	0.9795	0.0205	0.8430
Vídeo 10	0	210	5	0	NA	NA	0.9767	0.0233	0.9767
Vídeo 11	0	150	4	0	NA	NA	0.9740	0.0260	0.6438
Vídeo 13	0	338	5	0	NA	NA	0.9854	0.0146	0.9854
Vídeo 14	0	260	2	0	NA	NA	0.9924	0.0076	0.9924
Vídeo 16	0	152	5	0	NA	NA	0.9682	0.0318	0.9682
Vídeo 17	0	92	8	0	NA	NA	0.9200	0.0800	0.9200
Vídeo 18	4	92	1	103	0.8000	0.0374	0.9892	0.0108	0.4800
Valores medios	**4**	**118.7**	**1.71**	**103.14**	**0.7252**	**0.0643**	**0.9864**	**0.0130**	**0.5847**

Tabla 4-5: Resultados obtenidos en la evaluación del sistema con los *datasets* disponibles: segunda opción. Fuente: Propia

Se procede a comparar en el siguiente gráfico los resultados obtenidos para cada versión del algoritmo:

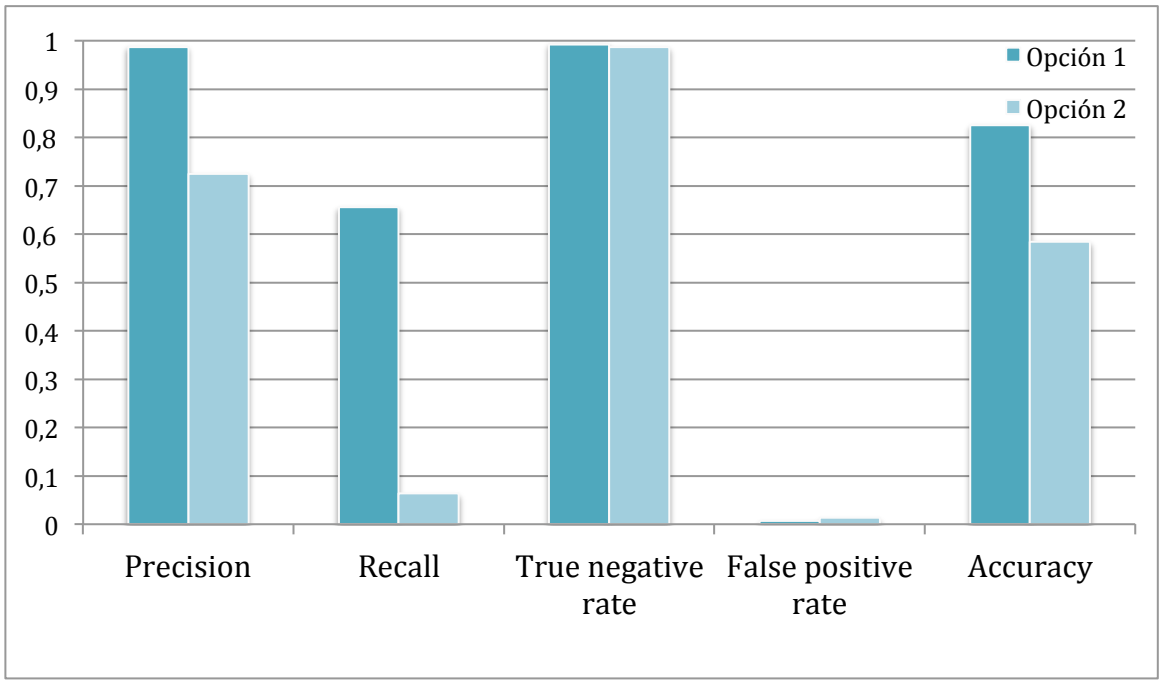

Figura 4-1: Comparación de las dos opciones propuestas para evaluar el sistema. Fuente: Propia.

Según estos resultados, podemos observar que el algoritmo tiene un comportamiento muy aceptable, ya que detecta caídas con bastante precisión, aunque suele depender en gran parte del vídeo leído. Comparando las dos opciones propuestas, vemos que los valores son similares para casi todos los casos, lo que indica que el sistema está funcionando bien, y que no "sobrepuntúa" en exceso, lo que supone un porcentaje de aciertos elevado en la detección de caídas. Sin embargo, vemos que esto no se cumple para el *recall,* donde sí se aprecia una notable diferencia entre ambas opciones. Esto es porque el número de falsos negativos que se producen aumenta para la opción 2, es decir, en varias ocasiones se califica como "no caída" una acción de "caída".

A continuación se procede a analizar detenidamente cada uno de los vídeos, describiendo sus características particulares y, dadas éstas, dónde reside la complejidad para detectar caídas.
Por ejemplo, los **vídeos 1, 2, 3, 6 y 7** presentan escenas con un conjunto de caídas consecutivas desde distintas posiciones cada una de ellas, combinadas con acciones de caminar y pasear por la estancia.
El sistema es muy preciso en todos estos casos, incluso en el del **vídeo 6** que, aunque en algún momento la persona sufre oclusión por algún objeto de la sala, no influye negativamente en el algoritmo.

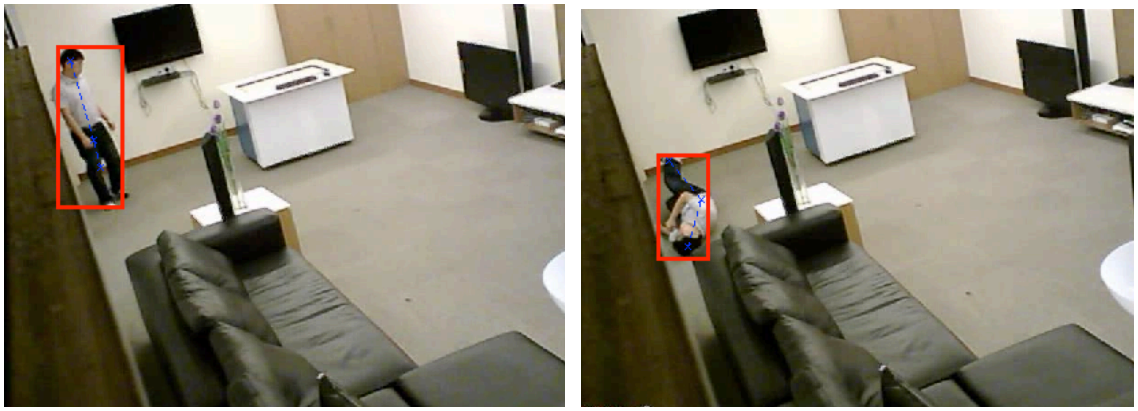

Figura 4-2: *Frames* del vídeo 6 con estados normales y de caída. Fuente: Propia.

Por otra parte, los **vídeos 9, 10 y 11** corresponden a escenas en las que el sujeto simula acciones que pueden llevar a detectar falsos positivos, como ponerse en cuclillas.
Observando el **vídeo 9** en particular, se ve que existe una caída menos fuerte que en otros de los casos, ya que el sujeto se queda de rodillas apoyándose con las manos en el suelo, uno de los posibles casos que podrían ocurrir si una persona mayor sufre un mareo o una bajada de tensión, en la que se desestabiliza y permanece durante un período de tiempo en dicha posición hasta que se recupera. En esta ocasión, se detecta como caída, que estrictamente no lo es, pero tampoco ha de considerarse como un suceso normal sin ningún tipo de alarma. Por esta razón, en mejoras futuras quizá podría establecerse un nuevo estado en este tipo de alarmas.

En este conjunto de vídeos mencionados se realizan acciones como agacharse en cuclillas, una actividad realizada de forma cotidiana, por ejemplo, al recoger algún objeto del suelo. En este caso, un sistema que funcionara de manera óptima no debería calificar dicha acción como caída y, sin embargo, nuestro sistema algunas las detecta de tal modo, debido al cambio producido entre las dos líneas extraídas como parámetros que se utilizan en la detección, sobre todo si este cambio se produce de forma brusca y repentina. Por lo tanto, esta es otra línea posible a incluir en el siguiente capítulo de trabajo futuro.

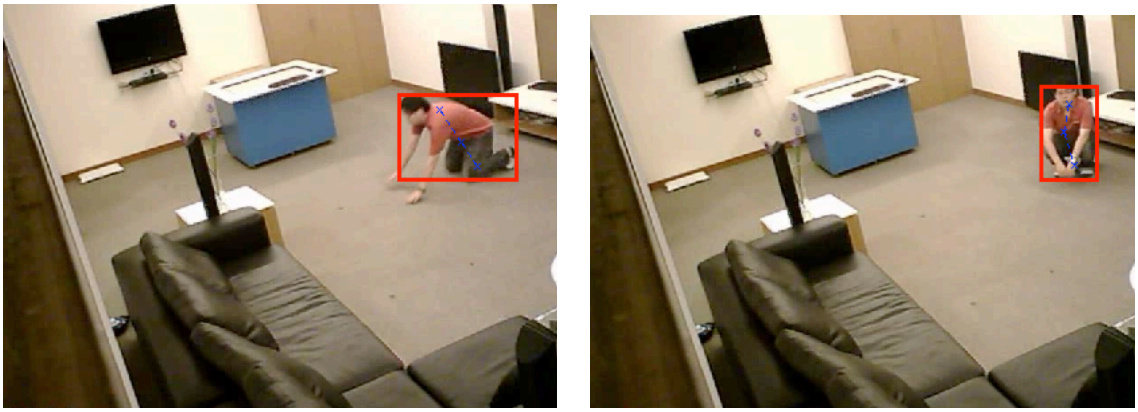

Figura 4-3: *Frames* del vídeo 9 con falsos positivos. Fuente: Propia

Dentro de los vídeos en los que no existe ningún tipo de caída, como los **vídeos 13 y 14**, para el **14** en concreto el sujeto aparece en la escena corriendo de un lado para otro y dando cambios bruscos de dirección, en los que en ciertos casos se detecta caída cuando no debería, dando lugar a algunos falsos positivos. Aunque habría que mejorarlo en un futuro, es un caso extremo que no causa demasiado trastorno, ya que normalmente una persona anciana, a quienes está enfocado principalmente este sistema, no va a realizar este tipo de acciones en su hogar.

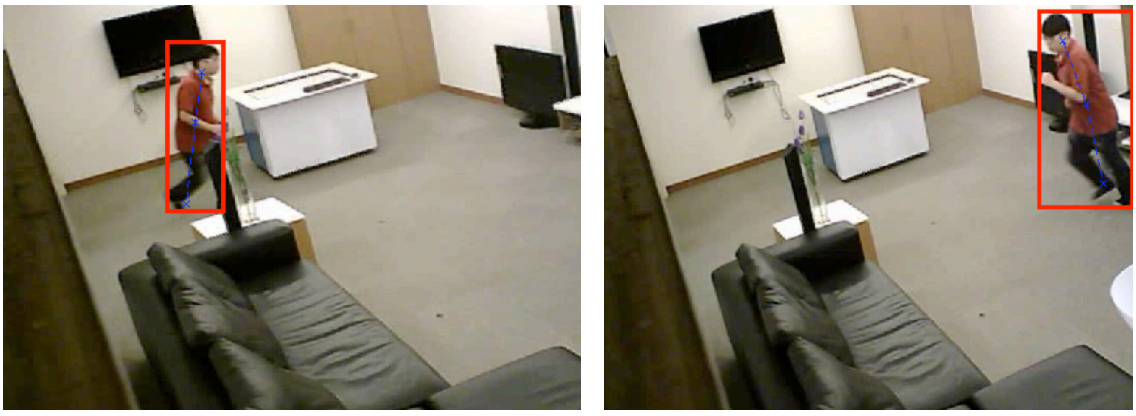

Figura 4-4: *Frames* del vídeo 14 con cambio brusco que no es caída. Fuente: Propia

Otro caso interesante de analizar es en los **vídeos 16 y 17**, en el cual se producen falsos positivos. Si bien es cierto que se produce un cambio brusco en las longitudes de las dos porciones en las que se divide la figura humana y que, por tanto, según nuestro algoritmo, se toma como una caída potencial, en este caso no se cumple que lo sea. Se trata de dos

movimientos repentinos en los que el sujeto se deja caer bruscamente en un sillón, como podemos ver en la siguiente figura:

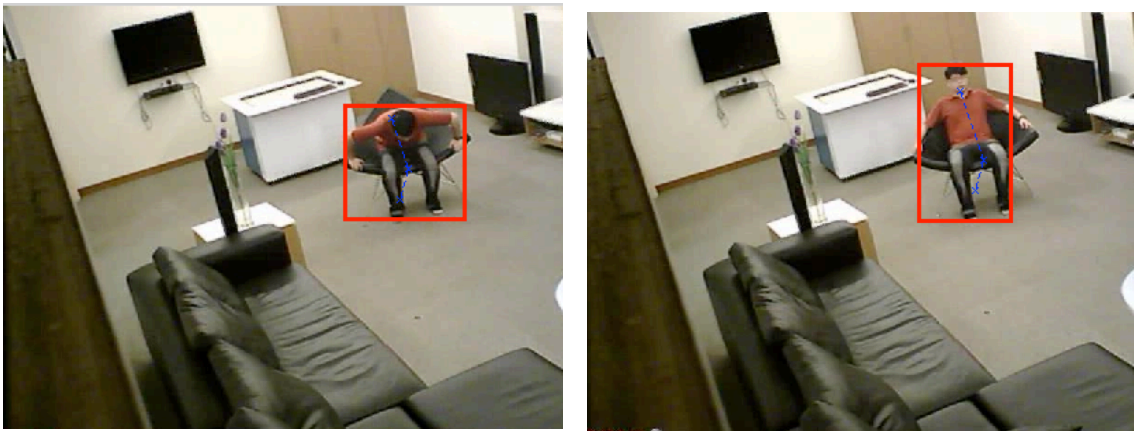

Figura 4-5: *Frames* del vídeo 16 con cambio brusco que no es caída. Fuente: Propia

Este sería otra de las principales líneas de mejora para el trabajo futuro. Aún así, se consigue un buen rendimiento del sistema, que detecta el resto de actividades normales como un estado normal, sin caída.

Por su parte, el **vídeo 18** aúna tanto falsos positivos, por sentarse el sujeto bruscamente en un sillón como en los que se acaba de comentar, con caídas detectadas correctamente. Además, el tipo de caída que se muestra es diferente hasta las ahora analizadas, ya que se produce directamente al intentar levantarse del sillón, contemplando así otro posible caso.

4.3.2 Coste computacional

La implementación de este algoritmo debería funcionar en tiempo real, es decir, mientras que un vídeo graba escenas de un hogar donde una persona realiza su vida de forma cotidiana, de modo que será posible localizar cualquier acción peligrosa que pueda llevar a una caída. Esto no ha sido posible, por lo que se ha optado por utilizar vídeos pregrabados de una duración limitada, alrededor de 20-25 segundos y, así, analizar dichas secuencias para detectar sucesos como caídas.
Por ello, calculando el tiempo de ejecución de todo el programa; esto es, desde que se lee el vídeo inicial, incluyendo el modelado de fondo, la extracción de los parámetros de la figura humana (estos dos son, con mucha diferencia, los que más tiempo consumen), el propio proceso de detección y también el cálculo de la métrica que nos permite observar las características de nuestro sistema, se tiene:

	Tiempo de ejecución por *frame* (segundos)	**Tiempo de ejecución de *frames* por segundo (fps)**
Vídeo 1	0.72	1.38
Vídeo 2	0.79	1.26
Vídeo 3	0.76	1.31

Vídeo 6	0.82	1.21
Vídeo 7	0.97	1.03
Vídeo 9	0.87	1.14
Vídeo 10	0.81	1.23
Vídeo 11	0.95	1.05
Vídeo 13	0.78	1.28
Vídeo 14	0.59	1.69
Vídeo 16	0.75	1.33
Vídeo 17	0.85	1.17
Vídeo 18	0.83	1.20
Valor medio	**0.87**	**1.35**

Tabla 4-6: Tiempos de ejecución del algoritmo. Fuente: Propia

Como se puede observar, el tiempo medio de ejecución por *frame* es 0.87 segundos. De este tiempo, alrededor de 0.20 segundos es lo que se dedica exclusivamente a la propia detección de la caída, ya que como se ha indicado, lo que mayor tiempo consume es la segmentación y extracción de parámetros. Es decir, para la detección del evento de interés producido se dedica únicamente un **22%** del tiempo total empleado.

4.3.3 Casos especiales

Para finalizar, existen una serie de casos particulares que deben mencionarse y tenerse en cuenta a la hora de evaluar el algoritmo implementado.

Tal y como está desarrollado el sistema, cuando se detecta una caída se acude a la 150 *frame* posterior y se decide si el sujeto sigue en el suelo o si por el contrario se ha levantado. Se daba este valor considerándolo un tiempo mínimo de espera (5 segundos), no pudiendo dar alguno mucho más mayor por disponer de *datasets* de entorno a los 20 segundos, ya que con saltos mayores se perderían muchas *frames* sin analizar.

Dado este diseño puede surgir un problema cuando el sujeto, una vez se cae, se levanta en menos de este período de 5 segundos establecido y vuelve a caerse. En este caso, se detectará una única caída y se determinará que el sujeto ha seguido inmóvil durante este tiempo, ya que al analizar la *frame* tras el salto así se encontrará, aunque no necesariamente por la causa pensada en el diseño, que correspondería a la primera caída.

4.4 *Conclusiones*

Tras las pruebas realizadas con los *datasets* disponibles y analizando cada uno de los casos particulares, se observa que el sistema ofrece un buen rendimiento en condiciones de caída combinadas con actividades simples y cotidianas, como caminar por una habitación, detectando caídas en la mayoría de los casos con un alto porcentaje de aciertos.

Sin embargo, cuando el individuo realiza actividades de otra índole, como agacharse en cuclillas o dejarse caer bruscamente en un sillón, el sistema creado puede confundir las características de esas acciones con las de caídas. Estas confusiones se deben a que los

parámetros y el cambio de éstos, que el sistema se encarga de analizar y los cuales tiene en cuenta a la hora de detectar una caída, son muy similares y varían de forma muy parecida entre este tipo de actividades.

Por tanto, se concluye que el sistema creado es robusto a la detección de caídas en situaciones simples, debiendo mejorarse para aquellas que llevan a confusión.

5 Conclusiones y trabajo futuro

5.1 *Conclusiones*

Este proyecto se basaba en la creación de un algoritmo que fuera capaz de detectar caídas simuladas en un entorno similar a uno doméstico.

Para este fin, se llevó a cabo un estudio de los distintos métodos del estado del arte, buscando aquella solución que mejor se adaptara a lo que se pretendía realizar. Esto es, un sistema sencillo pero robusto, basado en un análisis de la figura humana [7], una implementación interesante y diferente a otras que meramente se basaban en el análisis de cambio de la *bounding-box* [2][3][4][6]. Se buscaba un sistema que ofreciera unos resultados aceptables con un bajo coste computacional.

A partir de los resultados obtenidos y su posterior evaluación, se han podido establecer una serie de conclusiones principales:

- Para cualquier sistema basado en el procesado de vídeo es totalmente imprescindible una buena segmentación y un buen modelado de frente-fondo, ya que la bondad de todo el sistema que se genere a continuación dependerá en gran medida de éste.
- En relación con el punto anterior, hay que mencionar que las condiciones de captura del vídeo resultan factores críticos para el correcto funcionamiento del algoritmo. A mayor calidad y resolución, el ruido introducido en la imagen será menor, mejorando la segmentación y, como ya se ha comentado, el sistema en general.
- El enfoque proporcionado por el algoritmo implementado basado en el análisis de la figura humana funciona bastante bien en condiciones normales, detectando caída si ésta se combina con acciones como andar o caminar. Sin embargo, con aquellas que realizan movimientos más bruscos y repentinos, se observa que el sistema introduce falsos positivos, por lo que habría que hacerlo más robusto frente a estas situaciones.

En vista de los expuesto en estas líneas y a modo de conclusión global:

Se ha implementado un algoritmo de detección de caídas basado en el procesado de vídeo, partiendo de cero y realizando progresivamente todos los pasos necesarios para conseguirlo. Tras las pruebas realizadas y a la luz de los resultados obtenidos en las mismas, se afirma que se ha desarrollado un algoritmo capaz de detectar caídas en un entorno similar a uno doméstico con una razonable precisión.

5.2 *Trabajo futuro*

Dada la creciente esperanza de vida registrada cada año en la mayor parte del mundo, se cree necesario, a la par que muy útil, disponer de un sistema que permita a las personas ancianas que todavía se encuentran bien para desarrollar su vida con normalidad, vivir de manera independiente, contando siempre con un sistema de seguridad que les proporcione confianza y les haga sentir protegidos.
Es por ello que a este trabajo se le ve un gran potencial, con numerosas aplicaciones posibles. Por tanto, sería necesario seguir trabajando en este ámbito, intentando mejorar todas las características que lo debilitan o no lo hacen lo suficientemente robusto.

De manera que, las líneas de trabajo futuro principales en las que se considera que se podría profundizar más a partir de la idea expuesta son:

- ***Eficiencia:***

  Se debería conseguir resultados aún más eficientes que los obtenidos, aplicando para ello mejores y más complejas técnicas de segmentación, que hará mejorar todo el sistema al completo, así como profundizar más en las diferencias existentes en aquellas actividades que no son caída pero que confunden al algoritmo a la hora de detectarlas.
  Todo esto podría trasladarse a una implementación en C/C++ junto con la integración en un sistema de cámaras funcionando a tiempo real.

- ***Enfoques y aplicaciones:***

  Dada la duración finita de este proyecto, no se ha podido profundizar más en la búsqueda de otros enfoques, pero como trabajo futuro se tendría muy en cuenta introducir pequeños cambios en el algoritmo implementado, de modo que aceptase vídeos grabados mediante imágenes de profundidad.

  Como ya se analizó en el estado del arte, esta idea tiene muchas ventajas frente a aquellas basadas en vídeo convencional. Una de las principales es que el sujeto podría ser detectado igualmente en condiciones de muy baja luminosidad, lo que es mucho más práctico que un sistema que necesite cierta iluminación para detectar eventos.

Referencias

[1] United Nations, Department of Economic and Social Affairs, "*World Population Ageing 2013*", ST/ESA/SER.A/348; 2013.

[2] M. Mubashir, L.Shao, L. Seed, "*A survey on fall detection: Principles and approaches*", Neurocomputing; 100: 144-152; 2013.

[3] J. Willems, G. Debard, B. Bonroy, B. Vanrusmte, T. Goedemé, "*How to detect human fall in video? An overview*"; Proc. of the International Conference on Positioning and Context-Awareness, Antwerp; 2009.

[4] C. Rougier, J. Meunier, A. St-Arnaud, J. Rousseau, "*Robust Video Surveillance for Fall Detection Based on Human Shape Deformation*"; IEEE Transactions on circuits and systems for video technology; 21(5): 611-622; 2011.

[5] S.Patel, H. Park, P. Bonato, L. Chan, M. Rodgers, "*A review of wearable sensors and systems with application in rehabilitation*", NeuroEngineering and Rehabilitation, 1(9), 21; 2012.

[6] I. Charfi, J. Miteran, J. Dubois, M. Atri, R. Tourki, "*Definition and performance evaluation of a robust SVM based fall detection solution*", 8th International Conference on Signal Image Technology and Internet Based Systems; 218-224; 2012.

[7] J. Luen Chua, Y. Choon Chang, W. Keong Lim, "*A Simple Vision Based Fall Detection Technique for Video Surveillance*", Signal, Image and Video Processing, LNCS, 1-11; 2013.

[8] D. Anderson, R.H. Luke, J.M. Keller, M. Skubic, M. Rantz, M. Aud, "*Linguistic summarization of video for fall detection using voxel person and fuzzy logic*", Computer Vision and Image Understanding; 1(113), 80-89; 2009.

[9] D. H. Hung, H. Saito, "*The Estimation of Heights and Occupied Areas of Humans from Two Orthogonal Views for Fall Detection*", IEEJ Trans. EIS; 133(1), 1; 2013.

[10] E. Auvinet, C. Rougier, J. Meunier, A. St-Arnaud and J. Rousseau, "*Multiples cameras fall data set*", DIRO Université de Montréal, Tech. Rep, 1350; 2010.

[11] C. Rougier, E. Auvinet, J. Rousseau, M. Mignotte, J. Meunier, "*Fall Detection from Depth Map Video Sequences*", Springer Berlin Heidelberg, LNCS, 121-128; 2011.

[12] B. Kwolek, M. Kepski, "*Fall detection Using Kinect Sensor and Fall Energy Image*", Springer Berlin Heidelberg, LNCS, 294-303; 2013.

[13] M. Kepski, B. Kwolek,"*Unobtrusive Fall Detection at Home Using Kinect Sensor*", Springer Berlin Heidelberg, LNCS, 457-464; 2013.

[14] J. Davis, M. Goadrich, "*The relationship between Precision-Recall and ROC curves.*" Proceedings of the 23rd international conference on Machine learning. ACM, 233-240; 2006.

Anexo A

Inicialización (-1)
t = t + Tfondo

While frames en la secuencia de vídeo

if frame actual, anterior y Tsalto posterior son no nulas

$D1_t$, $D2_t$, $\theta1_t$, $\theta2_t$
$p_t = D1_t / D2_t$;

$D1_{t-1}$, $D2_{t-}$, $\theta1_{t-1}$, $\theta2_{t-1}$
$p_{t-1} = D1_{t-1}/ D2_{t-1}$;

$\Delta p = abs(p_t - p_{t-1})$;

No posible cambio:
No caída (0)
$\theta_r = \theta1_t$
$D_r = D1_t + D2_t$

if posible cambio, $\Delta p >= \varepsilon$
θ_{N1} = θ1{t+Tsalto}
θ_{N2} = θ2{t+Tsalto}
$\theta_{D1} = abs(\theta_{N1} - \theta_r)$;
$\theta_{D2} = abs(\theta_{N2} - \theta_r)$;
$\mu_\theta = mean(\theta_{D1} + \theta_{N2})$;
if $\mu_\theta > 30$
Caída (2)
else
$D_{diff} = abs(D_r - (D1\{t+Tsalto\} + D2\{t+Tsalto\}))$;
if D_{diff} > 40% de D_r;
Caída (2)
else
Falsa Alarma (1)
end
end
end

if **Caída (2)**
if t + Ttipocaida > numFrames – Tfondo
Guarda el estado de caída
t = numFrames - Tfondo
else
Guarda el estado de caída
t = t + Ttipocaida;
if frame en t no es nula
caida = check_inactivity(centroid{t-Ttipocaida}, centroid{t}) (3 o 4)
for frames del salto
No analizadas (-2)
end
end
end
end

Guarda el estado de caída
t = t + 1

end